mission research corporation

PSW FOREST AND RANGE
EXPERIMENT STATION

STATION LIBRARY COPY

WILDLAND FIRE MODELING FOR PLANNING SUPPORT

VOLUME I -- EXECUTIVE SUMMARY

VOLUME II -- INITIAL ATTACK ASSESSMENT MODELS

MRC Report No. 7611-3-876

Submitted to:

USDA Forest Service
Contract No. 21-393

Submitted by:

Mission Research Corporation
P.O. Drawer 719
Santa Barbara, CA 93102

Prepared by:

J. C. Sanderlin
J. M. Sunderson

August 1976

LIBRARY U.S. FOREST SERVICE BERKELEY

TABLE OF CONTENTS

STATION LIBRARY COPY

TABLE OF CONTENTS CONTINUED

TABLE OF CONTENTS CONTINUED

1.0 INTRODUCTION

This report is presented in four volumes, consisting of a summary volume (Volume I) and three volumes representing three of the four general specifications of USDA Forest Service Contract No. 21-393:

Volume II	--	Initial Attack Assessment Models (Specification 110)
Volume III	--	Segmented Perimeter Propagation Model (Specification 130)
Volume IV	--	Rate of Spread Model Assessment (Specification 140)

Specification 120 was satisfied by the submittal of two reports:

- "User's Guide: Experimental Initial Attack Evaluation Model," MRC Report No. 7611-1-376, March 1976,
- "Computer Code and Documentation for Experimental Initial Attack Evaluation Model," MRC Report No. 7611-2-376, March 1976,

which were submitted 6 months ARO as required.

2.0 INITIAL ATTACK ASSESSMENT MODELS

Volume two presents detailed analysis and documentation of the models employed for initial attack assessment. This information is included to provide potential users with deeper insight to theoretical and procedural considerations involved in particularly the models which simulate phenomenological functions.

3.0 SEGMENTED PERIMETER PROPAGATION MODEL

Volume III presents sample results from, and documentation of the segmented perimeter propagation model. Model results are compared with measured perimeters for the Potrero fire. The effects of modeling wildland fuel parameters, an alternative local directional spread rate algorithm, fire origin specification and perimeter point redistribution are examined.

4.0 RATE OF SPREAD MODEL ASSESSMENT

Volume IV presents the results of a coupled variables sensitivity analysis of the spread rate model. These results imply that wind velocity and fuel particle area-to-volume ratio are, respectively, the two most important input variables. It is suggested that the remaining variables can probably be modeled for fuel types of interest as a function of fuel age.

Because of the large dynamic variation of wind velocity in both space and time, wind velocity must be provided by a model capable of exhibiting both spatial and temporal effects, and ideally, with a predictive capability. Topographical data and to a large extent meteorological data will be determined by wind model input requirements to satisfy specified wind velocity accuracy expectations.

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES CONTINUED

LIST OF FIGURES CONTINUED

LIST OF FIGURES CONTINUED

LIST OF FIGURES CONTINUED

1.0 INTRODUCTION

The initial attack assessment model described in this volume was delivered, together with required documentation, 6 months ARO per USDA Forest Service Contract 21-393. The contents of this volume are a considerable elaboration of the minimum required documentation, and are intended to provide the user with additional information, particularly regarding theoretical and procedural considerations in important models.

2.0 RESOURCE CHARACTERISTICS DATA FILE CREATION (SLOAD)

1. Purpose

This routine and the associated block data routine are used to store resource characteristic data on a file called RCDATA.

2. Arguments

No arguments are required as input, and operation is fully automatic. To change resource characteristic data, the block data source must be changed and the program must be re-complied and re-loaded.

3. Procedure

Data are written from the common block where they were stored during compilation by the block data routine.

4. Comments

This program automatically chains to the loaded version of EIAEP.

5. Flow Chart

Not required.

```
$$$$$$LOAD
1030       COMMON/RSDATA/DAT(372)
1040       OPENFILE "RCDATA"("SYSTEM")
1050       WRITE("RCDATA")DAT
1055       CLOSEFILE "RCDATA"
1056C      PRINT,"USING EIAEP2L"
1060       CHAIN "EIAEP2L"("SYSTEM")
1070       END
6720       BLOCK DATA
6730C
6740C      THIS ROUTINE LOADS THE RESOURCE CHARACTERISTICS COMMON BLOCK
6750C      VERSION 1.10      1/26/76
6760C
6770       COMMON /RSDATA/ DECODI(2,8),MAXR,MAXS,MAXC,NAME(30),IDENT(15),
6780     &                 TRAVON(15),TRVOFT(15),TRVOFR(15),TRVAIR(15),
6790     &                 RADT(15),RADR(15),GRDT(15),GRDR(15),SUPVOL(15),
6800     &                 FULVOL(15),FULCON(15),PUMP(15),HOSCP1(15),
6810     &                 HOSCP2(15),HOSLN1(15),HOSLN2(15),MBLPMP(15),
6820     &                 PERSNL(15),SHIFTM(15),WRMAX(15),
6830     &                 DZBL(2),HCPE(3),ADPW(2)
6840C
6850       INTEGER DECODI,MAXR,MAXC,MAXS,NAME,IDENT,MBLPMP,PERSNL
6860C
6870       DATA DECODI/102,304,506,0,708,0,900,0,1000,0,1100,0,1213,0,1415,0/
6880       DATA MAXR,MAXS,MAXC/15,4,8/
6890C
6900       DATA NAME/4HENG1,4HENG2,4HENG3,4HENG4,4HHCT1,4HHCT2,4HDOZ1,4HDOZ2,
6910     &           4HRWHC,4HFWRD,4HRWWD,4HFULR,4HFULR,4HTK1 ,4HTK1 ,
6920     &           8*4H    ,5*4H1    ,2*4H    /
6930       DATA IDENT/1001,1002,1003,1004,2001,2002,3001,3002,4001,5001,6001,
6940     &            7001,7002,8001,8002/
6950       DATA TRAVON/1341.,1341.,1341.,1475.,1341.,1475.,1072.,1072.,3*0.,
6960     &             938.,1072.,938.,1072./
6970       DATA TRVOFT/0.,0.,134.1,402.3,0.,402.3,9*0./
6980       DATA TRVOFR/6*0.,107.3,107.3,7*0./
6990       DATA TRVAIR/15*0.0/
7000       DATA RADT/9.144,7.620,7.620,4.572,9.144,457.2,30.48,30.48,5*0.,
7010     &           9.144,7.620/
7020       DATA RADR/6*0.,2.438,3.048,7*0./
7030       DATA GRDT/20.,20.,20.,30.,25.,30.,10.,10.,3*0.,20.,30.,20.,20./
7040       DATA GRDR/6*0.,50.,50.,7*0./
7050       DATA SUPVOL/2839.,1893.,757.1,189.3,5*0.,7571.,1041.,3785.,378.5,
7060     &             7571.,3785./
7070       DATA FULVOL/132.5,132.5,75.71,75.71,75.71,113.6,227.1,113.6,7*0./
7080       DATA FULCON/0.5322,0.5035,0.3785,0.3142,0.3142,0.3785,0.2271,
7090     &             0.3142,7*0./
7100       DATA PUMP/4732.,3785.,1893.,56.78,9*0.,1329.,56.78/
7110       DATA HOSCP1/283.9,283.9,283.9,56.78,11*0./
7120       DATA HOSCP2/567.8,567.8,567.8,12*0./
7130       DATA HOSLN1/152.4,457.2,304.8,45.72,11*0./
7140       DATA HOSLN2/609.6,304.8,152.4,12*0./
7150       DATA MBLPMP/0,0,1,1,11*0/
7160       DATA PERSNL/5,5,5,2,16,16,2,2,8,6*2/
7170       DATA SHIFTM/600.,600.,600.,600.,600.,600.,600.,600.,3*0.0,600.,
7180     &             600.,600.,600./
7190       DATA WRMAX/16.76,14.02,8.839,3.353,0.3066,0.2453,11.13,10.06,
7200     &            0.3832,131.1,39.62,4*0./
7210       DATA DZBL/2.438,1.829/,HCPE/0.6,1.0,1.25/,ADPW/0.20,0.15/
7220C          COMMON BLOCK VARIABLE DEFINITIONS
7230C      DECODI - DECODING ARRAY
7240C      MAXR   - MAXIMUM NUMBER OF RESOURCES
7250C      MAXS   - MAXIMUM NUMBER OF SUBCLASSES
```

```
C       MAXC   - MAXIMUM NUMBER OF CLASSES
C       NAME   - HOLLERITH MNEMONIC
C       IDENT  - RESOURCE ID, THE FORM OF IDENT IS CTTT, WHERE C IS THE
C                CLASS AND TTT IS THE SUBCLASS
C       TRAVON - MAXIMUM ON-ROAD TRAVEL RATE(METER/MIN)
C       TRVOFT - TRANSPORT MAXIMUM OFF ROAD TRAVEL RATE(METER/MIN)
C       TRVOFR - RESOURCE MAXIMUM OFF ROAD TRAVEL RATE(METER/MIN)
C       TRVAIR - MAXIMUM AIR TRAVEL RATE(METER/MIN)
C       RADT   - TRANSPORT MINIMUM TURN RADIUS(METER)
C       RADR   - RESOURCE MINIMUM TURN RADIUS(METER)
C       GRDT   - TRANSPORT MAXIMUM GRADE CAPABILITY(PERCENT)
C       GRDR   - RESOURCE MAXIMUM GRADE CAPABILITY(PERCENT)
C       SUPVOL - SUPPLY VOLUME, WATER OR FUEL(LITER)
C       FULVOL - FUEL TO RUN ON, VOLUME(LITER)
C       FULCON - FUEL CONSUMPTION RATE(LITER/MIN)
C       PUMP   - PUMP CAPACITY(LITER/MIN)
C       HOSCP1 - 2 1/2 INCH HOSE PUMPING CAPACITY(LITER/MIN)
C       HOSCP2 - 1 1/2 INCH HOSE PUMPING CAPACITY(LITER/MIN)
C       HOSLN1 - 2 1/2 INCH HOSE LENGTH(METER)
C       HOSLN2 - 1 1/2 INCH HOSE LENGTH(METER)
C       MBLPMP - MOBILE CAPABILITY WHILE PUMPING(YES=1,NO=0)
C       PERSNL - NUMBER OF PERSONNEL
C       SHIFTM - WORK SHIFT TIME(MINUTES)
C       WRMAX  - MAXIMUM WORK RATE(UNITS DEPEND ON RESOURCE)
C
C    NOTE - THE RESOURCE CHARACTERISTICS COMMON WILL NORMALLY APPEAR IN
C           THE FOLLOWING FORM
C
C       COMMON /RSDATA/ DECODT(2,8),MAXR,MAXS,MAXC,RESCHR(15,23),AUX(8)
C
        END
```

3.0 EVALUATE INITIAL ATTACK (EIAEP)

1. Purpose

This routine performs most of the initial attack evaluation interaction and sequences all required calculations.

2. Input Requirements

With the exception of one file, all inputs are interactively obtained from the user. The resource characteristic data file (RCDATA) is created automatically by another program (SLOAD) and read by the EIAEP. All other interactive inputs are described in "User's Guide: Experimental Initial Attack Evaluation Model."[1]

3. Procedure

Input data are obtained from the user interactively and various models are invoked to calculate perimeters, to calculate probabilities of success and to print the results. The models used are described elsewhere in this document.

4. Comments

The listings which follow the flow chart are for four different programs (S2EIAEP, MP1, MP2, RS). The EIAEP described here was broken into these four parts for overlay purposes. Logically the four parts should be viewed as a unit.

5. Flow Chart

See Figure 1.

Figure 1 Evaluate Initial Attack

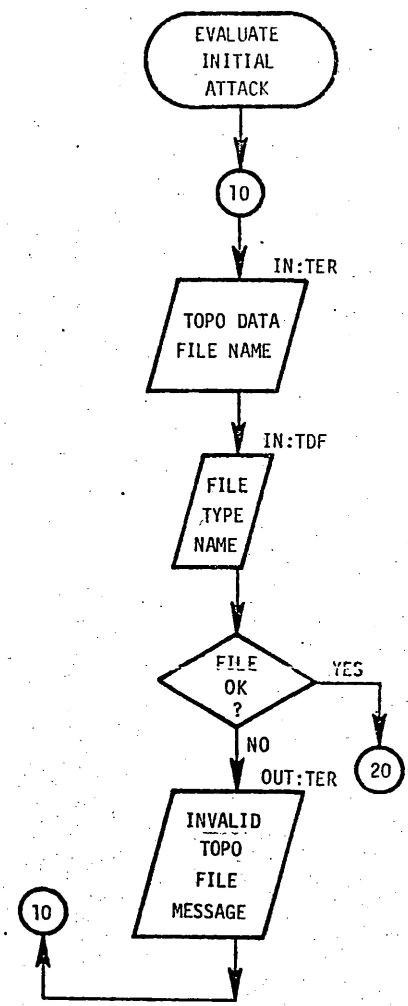


Figure 1 Evaluate Initial Attack (Cont'd)

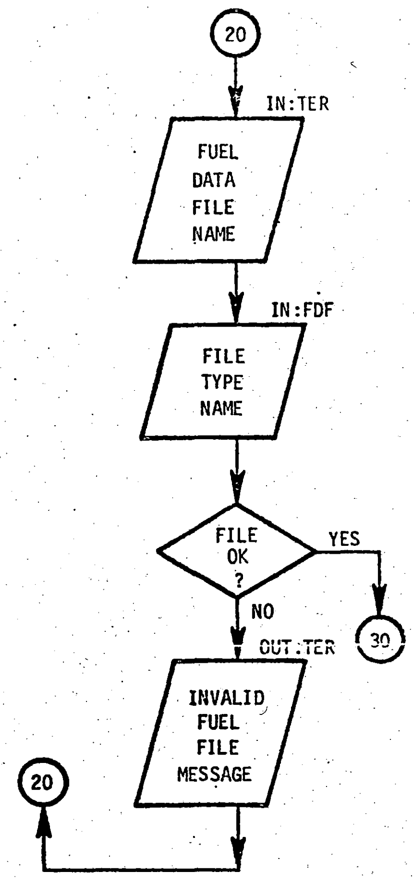


Figure 1 Evaluate Initial Attack (Cont'd)

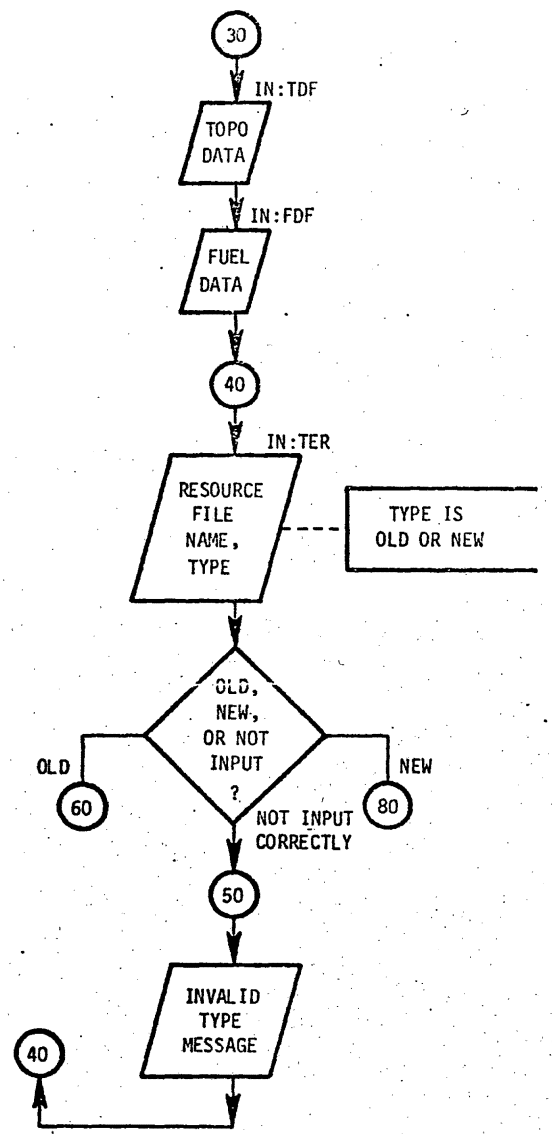


Figure 1 Evaluate Initial Attack (Cont'd)

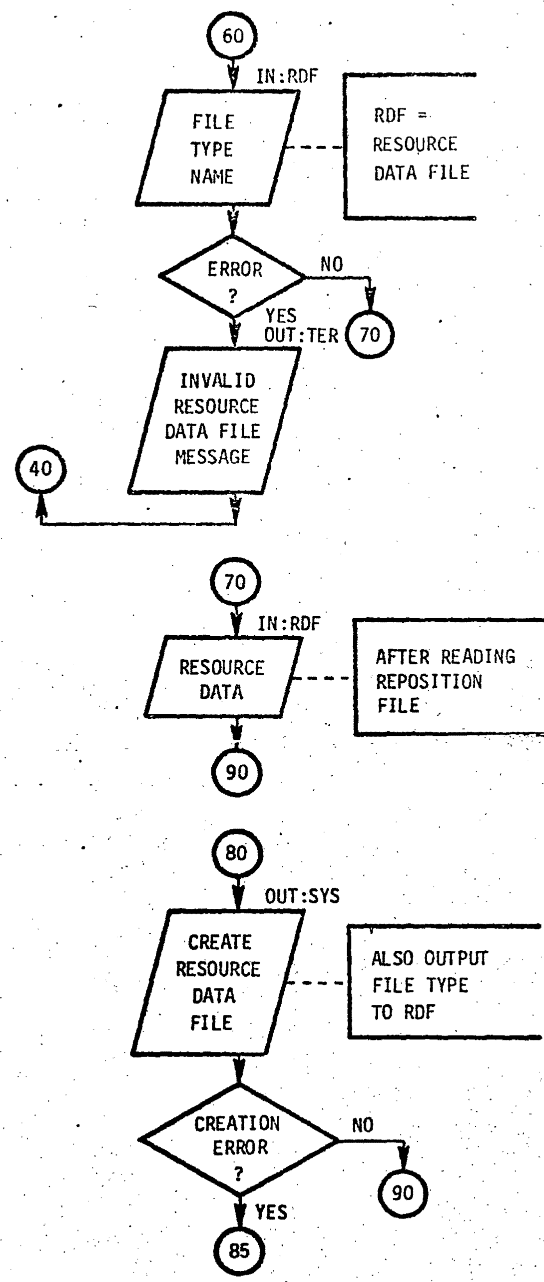


Figure 1 Evaluate Initial Attack (Cont'd)

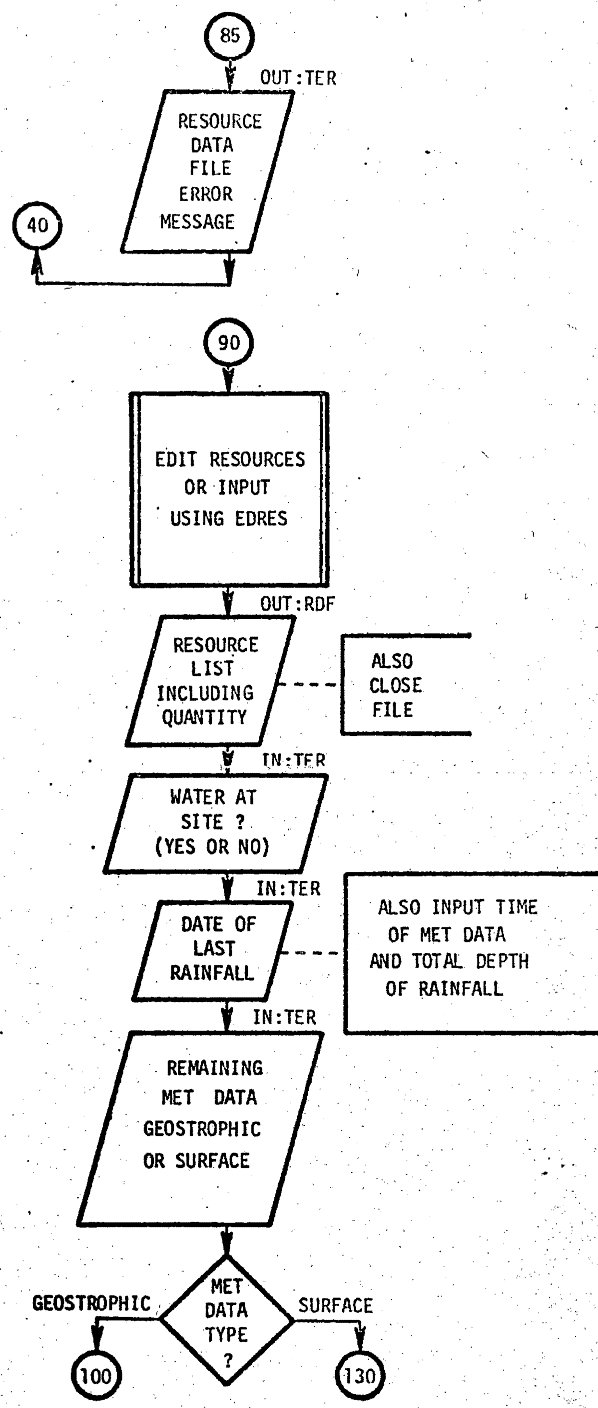


Figure 1 · Evaluate Initial Attack

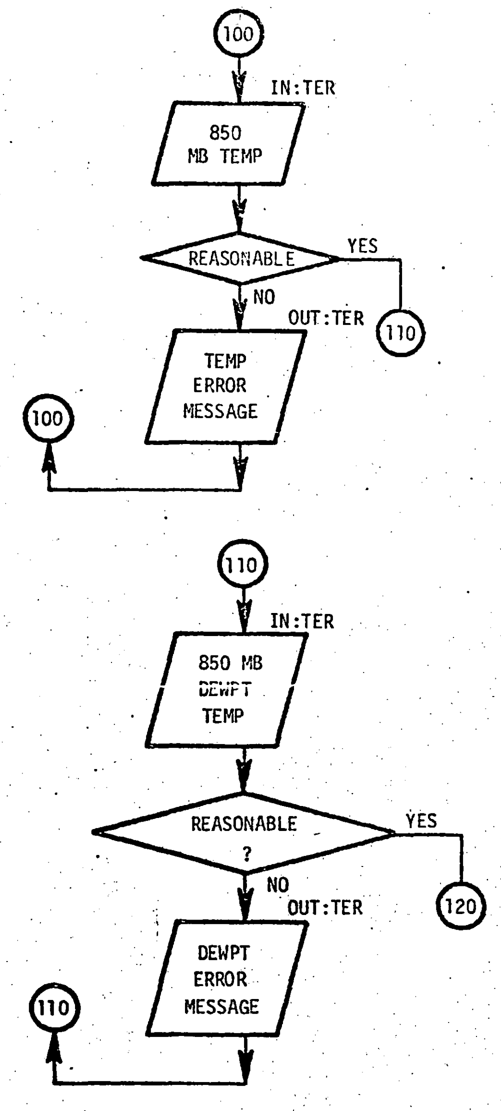


Figure 1 Evaluate Initial Attack (Cont'd)

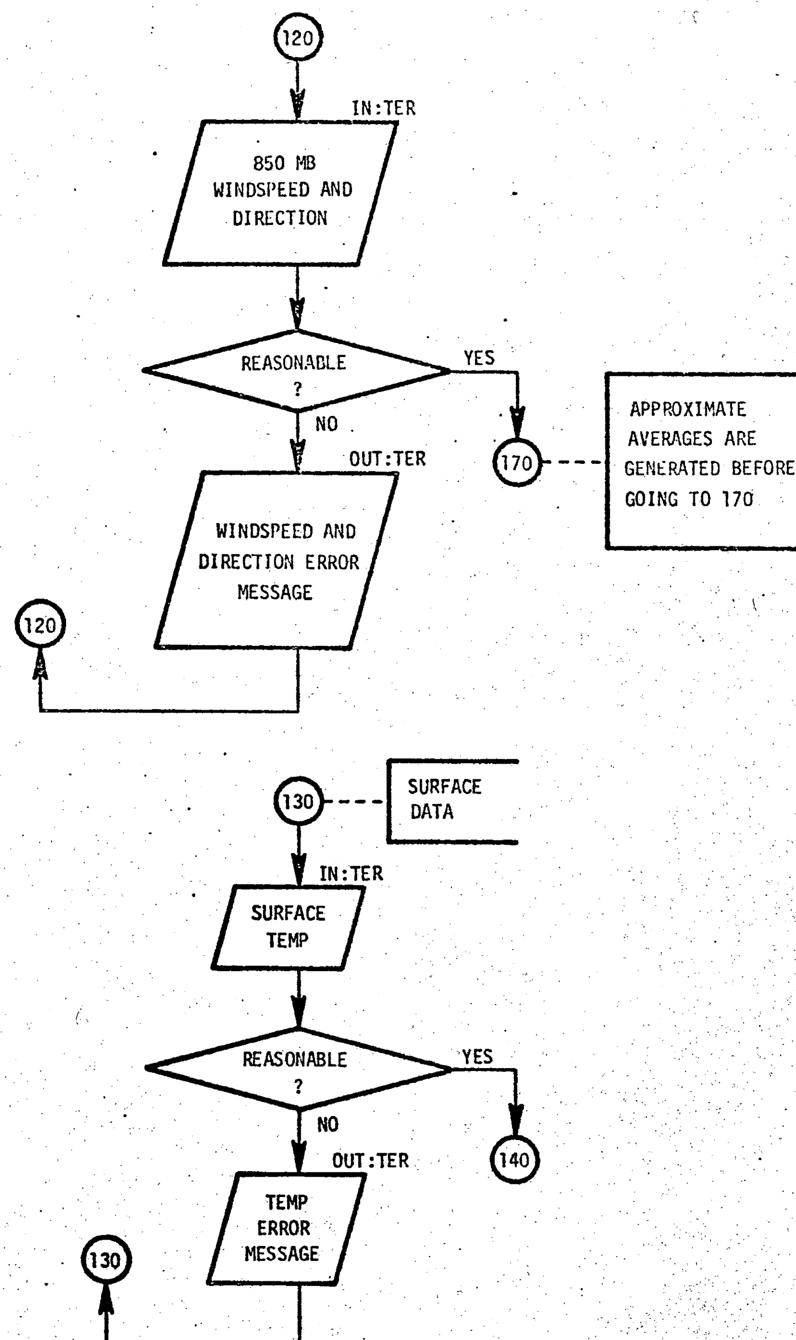


Figure 1   Evaluate Initial Attack (Cont'd)

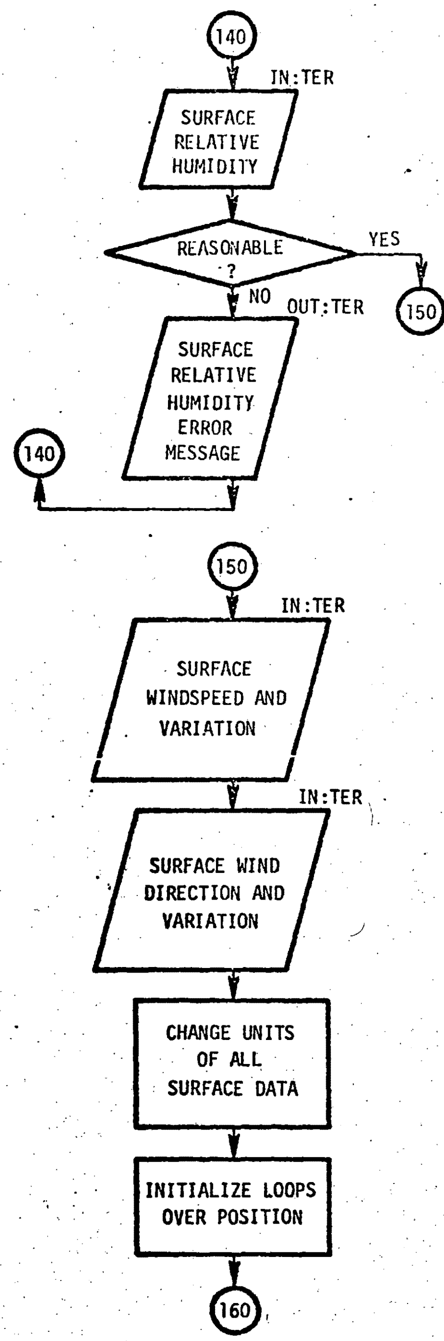


Figure 1 Evaluate Initial Attack (Cont'd)

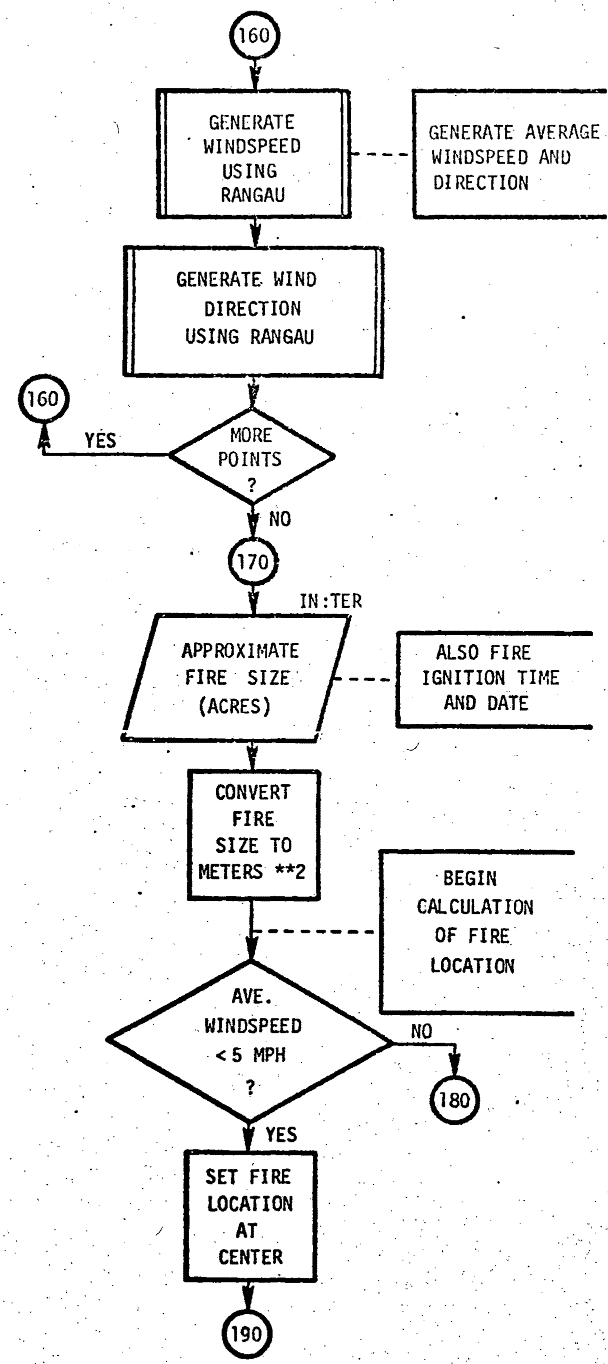


Figure 1 Evaluate Initial Attack (Cont'd)

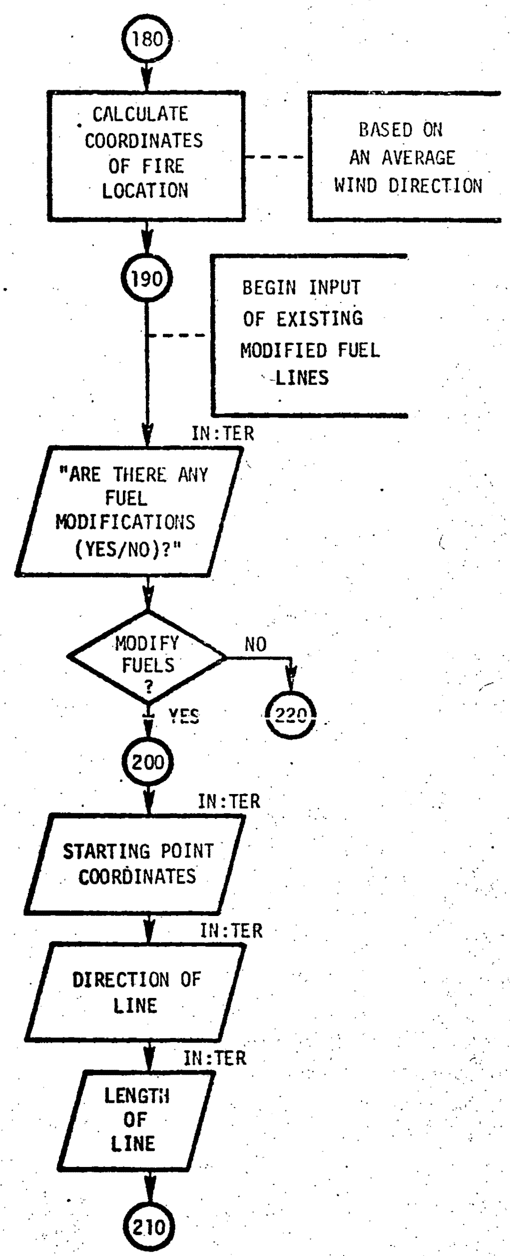
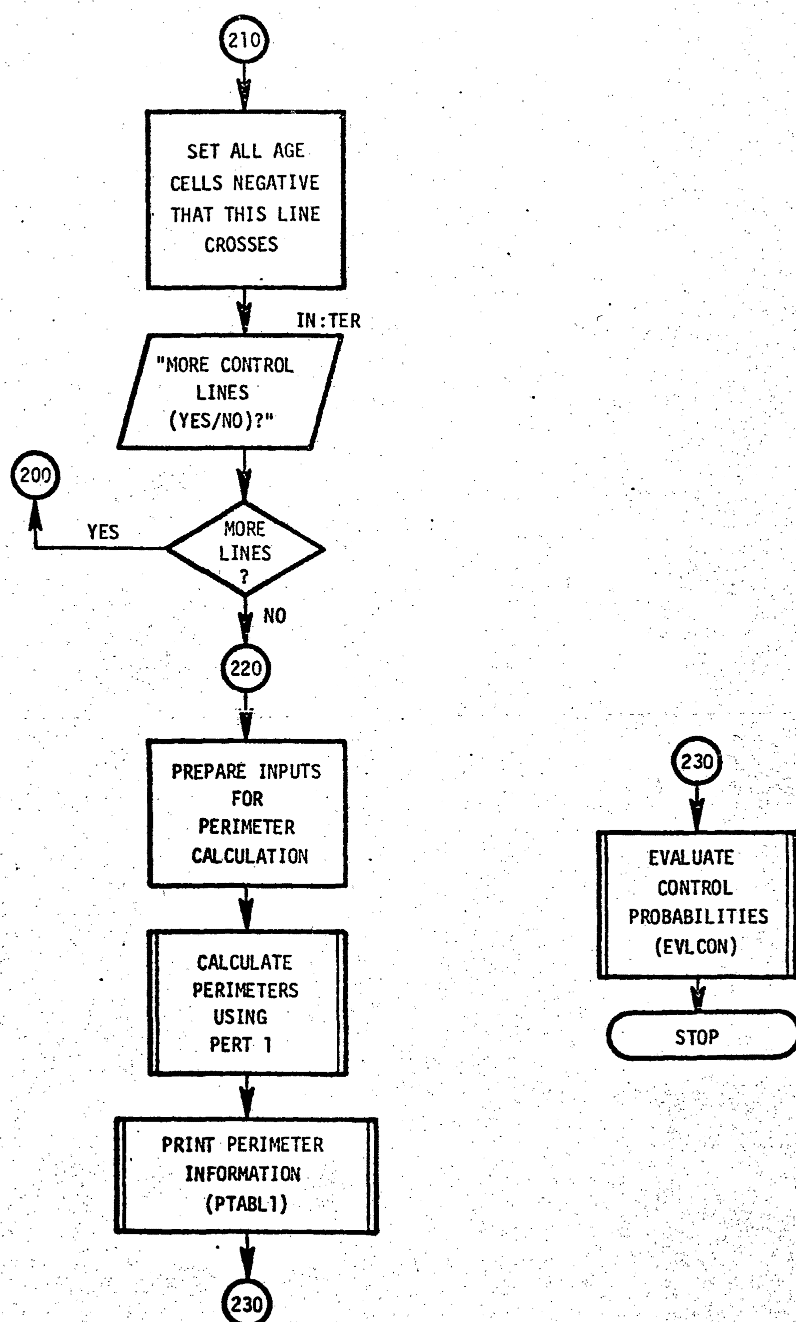


Figure 1 Evaluate Initial Attack (Cont'd)



```
$$$$$S2EIAEP
C      OPTION MAP
       $USE FILE1,FILE3,FILE4,RS,MP1,MP2,EVLP1,EVLP2
       OPTION LOAD
       OPTION NOLINE,NOCHECK
C
C      EXPERIMENTAL INITIAL ATTACK EVALUATION PROGRAM (TO BE USED IN
C          EVALUATING THE POSSIBLE SUCCESS OR FAILURE OF AN INITIAL
C          ATTACK ON WILDLAND FIRES)
C      VER 3.00 (FOR USE ON G. E. TIMESHARE ONLY)
C
C      PROGRAMMER - JOHN M. SUNDERSON, JR.
C
       FILELIST TDF,FDF,RDF
C          FILES ARE AS FOLLOWS.......
C      TDF     - TOPO DATA FILE (PREPARED BY A SEPARATE PROGRAM)
C      FDF     - FUEL DATA FILE (PREPARED BY A SEPARATE PROGRAM)
C      RDF     - RESOURCES DATA FILE (CREATED AND ALTERED BY THIS PROGRAM)
C
C          ROUTINES USED
C      CKNAMF
C
       ALPHA WANS,GEOS,SURF
       ALPHA RDFNAM(3),NAMRDF(3),YES,NO,WATERA,ANS
       ALPHA TDFNAM(3),NAMTDF(3),FDFNAM(3),NAMFDF(3),RFLTYP,OLD,NEW
       ALPHA STOPP,CONT
C
       LOGICAL OK
       LOGICAL CKNAMF,NOTOK
C
       COMMON/TOPO2/KUADT,X0T,Y0T,XMT,YMT,DXTI,DXTII,DYTI,DYTII,SRUF,
      &                 DXT02,DYT02,NXTP,NYTP,NPTST,HBC(51,51)
       COMMON/RSDATA/DATRS(372)
       COMMON/FUEL2/KUADF,X0F,Y0F,XMF,YMF,DXFI,DXFII,DYFI,DYFII,
      &                 DXF02,DYF02,NXFC,NYFC,NCELF,AGET(50,50),IFTYPE
```

```
C
      COMMON/ROUT12/NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
     &                 ETA(20),ACT(20)
      COMMON/SPLMET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
      COMMON/MET37/WANS,T850C,DP850C,WSMPM8,WDR8,
     &                KUADW,XOW,YOW,XMW,YMW,DXWI,DXWII,DYWI,DYWII,
     &                DXWO2,DYWO2,NXWP,NYWP,NPTSW,WINDSG(7,7),WINDDG(7,7),
     &                DVSURR,WVSURM,RHSUR,TSURC
      COMMON/FIREI/XF,YF,AREA1,PTIN(11),NPTRQ,PTCAL(11),NPTCAL,IDISP
      COMMON/FMOD1/CMESH,TSPRED,ISPREP,METIN,CTHRSH
      COMMON/TIMEI/JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
      COMMON/PERIM1/NPT,NXY,DELT,PT(11),PX(20,11),PY(20,11)
      COMMON/CNSTNT/RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
      COMMON /IMPPER/ TINCR,NTSTEP,PL(51),WDN(51),PCNR(51),PCNT(51),
     &                  PGR(51),CWR(51)
      COMMON/MCON/IYFI,JFI
      COMMON /WARNS/ IWARN(3)
      COMMON/BREKIT/NSTEP,IQSTEP,IMSTEP,IPSTEP(12),
     & ICFLAG(12),IMMSTP(12)
C
      DATA GEOS/4HGEOS/,SURF/4HSURF/
      DATA NAMRDF/4HRESO,4HURCE,4H-DAT/,YES/3HYES/,NO/2HNO/
      DATA OLD/3HOLD/,NEW/3HNEW/
      DATA NAMTDF/4HTOPO,4H-DAT,1HA/,NAMFDF/4HFUEL,4H-DAT,1HA/
      DATA STOPP/4HSTOP/, CONT/4HCONT/
C
      CALL BREAK(0)
C
C     LOAD RESOURCE DATA
C
      CALL LLINK("ORS")
      CALL RS
      CALL IFBREAK($1000)
      CALL BREAK(1)
C
C     INITIALIZE FOR A NEW FIRE
C
   10 CALL BREAK(0)
      IWARN(1)=1
      IWARN(2)=1
      IWARN(3)=0
      NSTEP=12
      DO 120 I=1,NSTEP
      IPSTEP(I)=0
      ICFLAG(I)=0
      IMMSTP(I)=1
  120 CONTINUE
      DO 122 I=8,NSTEP
      IMMSTP(I)=3
  122 CONTINUE
      IMMSTP(7)=2
      IQSTEP=1
      CALL BREAK(1)
    1 CALL LLINK("UMP1")
      CALL MP1(TDF,FDF,RDF,RFLTYP,DISTRI,RDFNAM)
C
C     EDIT OR INPUT RESOURCES
C
    2 CALL LLINK("UFILE1")
      CALL FILE1
      CALL EDIRES(RFLTYP,IYFI)
    3 CALL LLINK("UMP2")
      CALL MP2(TDF,FDF,RDF,WATERA,RDFNAM)
C          CALCULATE PERIMETER
    4 CALL LLINK("UFILE3")
      CALL FILE3
```

```
      IWARN(3)=0
      CALL PERI1(IYFI)
    5 CALL LLINK("UFILE4")
      CALL FILE4
      CALL PTABL1(IYFI)
C
C     EVALUATE INITIAL ATTACK
C
    6 CALL LLINK("UEVLP1")
      CALL FEVLP1
      CALL EVLP1(IYFI,WATERA,DISTR1)
    7 CALL LLINK("UEVLP2")
      CALL FEVLP2
      CALL EVLP2(IYFI,WATERA)
C
      IPSTEP(IQSTEP)=1
  100 PRINT     2001
 2001 FORMAT(3X,45HDO YOU WISH TO EVALUATE ANOTHER FIRE (YES/NO))
      PRINT     2002
 2002 FORMAT(7X,4H....)
      READ     2003,ANS
 2003 FORMAT(A4)
      IQSTEP=1
      IF(ANS.EQ.YES)GO TO 10
      IF(ANS.EQ.NO)STOP
      PRINT     2004
 2004 FORMAT(3X,26H***ERROR*** INVALID ANSWER)
      GO TO 100
C
C     BREAK KEY HAS BEEN PRESSED
C
 1000 CALL IFBREAK($1000)
 1001 PRINT     2005
 2005 FORMAT(3X,30H*BREAK* ENTER STOP OR CONTINUE)
      PRINT     2002
      READ     2003,ANS
      IF(ANS.EQ.STOPP)STOP
      IF(ANS.EQ.CONT)GO TO 1002
      PRINT     2006
 2006 FORMAT(3X,25H***ERROR*** INVALID ENTRY)
      GO TO 1001
 1002 PRINT     2007
 2007 FORMAT(3X,17HENTER STEP NUMBER)
      PRINT     2002
      READ     2008,IQSTEP
 2008 FORMAT(V)
      IF(IQSTEP.GE.1 .AND. IQSTEP.LE.NSTEP)GO TO 1003
      PRINT     2009,NSTEP
 2009 FORMAT(3X,37H***ERROR*** INVALID STEP, MUST BE 1 -,I3)
      GO TO 1002
 1003 ISTPB=IQSTEP+1
      ILOOP=IQSTEP
      DO 1004 ISTP=1,ILOOP
      ISTPB=ISTPB-1
      IF(IPSTEP(ISTPB).LE.0)IQSTEP=ISTPB
 1004 CONTINUE
      DO 1005 ISTP=IQSTEP,NSTEP
      IPSTEP(ISTP)=0
 1005 CONTINUE
      IMSTEP=IMMSTP(IQSTEP)
      GO TO (1,2,3),IMSTEP
      END
```

```
$$$$$SMP1
1010        $ LINK UMP1
1025        SUBROUTINE MP1(TDF,FDF,RDF,RFLTYP,DISTRI,RDFNAM)
1030C
1040C       EXPERIMENTAL INITIAL ATTACK EVALUATION PROGRAM (TO BE USED IN
1050C           EVALUATING THE POSSIBLE SUCCESS OR FAILURE OF AN INITIAL
1060C           ATTACK ON WILDLAND FIRES)
1070C       VER 3.00 (FOR USE ON G. E. TIMESHARE ONLY)
1080C
1090C       PROGRAMMER - JOHN M. SUNDERSON, JR.
1100C
1110        FILENAME RDF,FDF,TDF
1120C           FILES ARE AS FOLLOWS.......
1130C       TDF     - TOPO DATA FILE (PREPARED BY A SEPARATE PROGRAM)
1140C       FDF     - FUEL DATA FILE (PREPARED BY A SEPARATE PROGRAM)
1150C       RDF     - RESOURCES DATA FILE (CREATED AND ALTERED BY THIS PROGRAM)
1160C
1170C           ROUTINES USED
1180C       CKNAMF
1190C
1195        ALPHA BLANK
1200        ALPHA WANS,GEOS,SURF
1210        ALPHA RDFNAM(3),NAMRDF(3),YES,NO,WATERA,ANS
1220        ALPHA TDFNAM(3),NAMTDF(3),FDFNAM(3),NAMFDF(3),RFLTYP,OLD,NEW
1230C
1240        LOGICAL OK
1250        LOGICAL CKNAMF,NOTOK
1260C
1270        COMMON/TOPO2/KUADT,X0T,Y0T,XMT,YMT,DXTI,DXTII,DYTI,DYTII,SRUF,
1280       &             DXT02,DYT02,NXTP,NYTP,NPTST,HBC(51,51)
1290        COMMON/FUEL2/KUADF,X0F,Y0F,XMF,YMF,DXFI,DXFII,DYFI,DYFII,
1300       &             DXF02,DYF02,NXFC,NYFC,NCELF,AGET(50,50),IFTYPE
1310C
1320        COMMON/ROUT12/NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
1330       &             ETA(20),ACT(20)
1340        COMMON/SPLMET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
1350        COMMON/MET3/WANS,T850C,DP850C,WSMP48,WDR8,
1360       &            KUADW,X0W,Y0W,XMW,YMW,DXWI,DXWII,DYWI,DYWII,
1370       &            DXW02,DYW02,NXWP,NYWP,NPTSW,WINDSG(7,7),WINDDG(7,7),
1380       &            DVSURR,WVSURM,RHSUR,TSURC
1390        COMMON/FIRE1/XF,YF,AREA1,PTIN(11),NPTRQ,PTCAL(11),NPTCAL,IDISP
1400        COMMON/FMOD1/CMESH,TSPRED,ISPREP,METIN,CTHRSH
1410        COMMON/TIME1/JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
1420        COMMON/PERIM1/NPT,NXY,DELT,PT(11),PX(20,11),PY(20,11)
1430        COMMON/CNSTNT/RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
1440        COMMON /IMPPER/ TINCR,NTSTEP,PL(51),WDN(51),PCNR(51),PCNT(51),
1450       &                PGR(51),CWR(51)
1455        COMMON/RSDATA/DECODI(2,8),MAXR,MAXS,MAXC,RESCHR(15,23),AUX(8)
1456        COMMON/MCON/IYFI,JFI
1458        COMMON/BREKIT/NSTEP,IQSTEP,IMSTEP,IPSTEP(12),
1459       & ICFLAG(12),IMMSTP(12)
1460C
1465        DATA BLANK/1H /
1470        DATA GEOS/4HGEOS/,SURF/4HSURF/
1480        DATA NAMRDF/4HRESO,4HURCE,4H-DAT/,YES/3HYES/,NO/2HNO/
1490        DATA OLD/3HOLD/,NEW/3HNEW/
1500        DATA NAMTDF/4HTOPO,4H-DAT,1HA/,NAMFDF/4HFUEL,4H-DAT,1HA/
1510C
1511        JUMPS=IQSTEP
1512        GO TO (1,2,40,90,92,172),JUMPS
1513      1 IF(ICFLAG(IQSTEP).GT.0)GO TO 10
1520C           FILL CONSTANT COMMON
1530        RE=6.37E6
1540        PI=3.14159265358979
1550        PIO180= 1.74532925E-2
1560        HALFPI=1.57079633
1570        TWOPI=6.28318531
1580        FOURPI=12.5663706l4
```

```
      ATEPI=25.1327412287
      GRAVZ=35303.94
      NRSMAX=20
      ICFLAG(IQSTEP)=1
C
C     SET PROMPT CHARACTER
C
      CALL SETCHR(32)
C
C     IDENTIFY PROGRAM AND OPERATIONS TO BE PERFORMED
C
      PRINT     1001
 1001 FORMAT(1X,61HEXPERIMENTAL INITIAL ATTACK EVALUATION PROGRAM - VERS
     &ION 3.00/)
C
C     INPUT FROM TERMINAL - TOPO DATA FILE NAME
C
   10 PRINT     1002
 1002 FORMAT(3X,28H1. ENTER TOPO DATA FILE NAME)
C
      PRINT     1003
 1003 FORMAT(7X,4H....)
C
      READ     2001,TDF
 2001 FORMAT(A8)
C
C     READ FILE TYPE FROM TDF
C
      CALL BREAK(0)
      READ(1,2002,END=12,ERR=12)TDFNAM
      IF(CKNAME(TDFNAM,NAMTDF,3))GO TO 20
      CLOSEFILE 1
   12 CALL BREAK(1)
      PRINT     1004,TDFNAM
 1004 FORMAT(3X,44H***ERROR*** WRONG OR MISSING FILE. LABEL IS ,3A4)
      GO TO 10
C
C     INPUT FROM TDF - TOPO DATA
C
C         READ DATA HEADER
   20 READ(1,2030,END=12,ERR=12)KUADT,XOT,YOT,XMT,YMT,DXTI,
     &    DXTII,DYTI,DYTII,SRUF,DXTO2,DYTO2,NXTP,NYTP,NPTST
 2002 FORMAT(3A4)
 2030 FORMAT(I5,4E15.7/4E15.7/ 3E15.7,2I5,I10)
C         READ TOPO DATA
      READ(1,2004,END=12,ERR=12)HBC
 2004 FORMAT(5E16.8)
      CLOSEFILE 1
      CALL CLOSER(XDUM)
C
C     INPUT FROM TERMINAL - FUEL DATA FILE NAME
C
      IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
      CALL BREAK(1)
    2 PRINT     1005
 1005 FORMAT(3X,28H2. ENTER FUEL DATA FILE NAME)
      PRINT     1003
      READ     2001,FDF
C
C     READ FILE TYPE FROM FDF
C
      FDFNAM(1)=TDFNAM(1)
      FDFNAM(2)=TDFNAM(2)
      FDFNAM(3)=TDFNAM(3)
      IF(FDF.EQ.TDF)GO TO 22
```

```
      FDFNAM(1)=BLANK
      FDFNAM(2)=BLANK
      FDFNAM(3)=BLANK
      CALL BREAK(0)
      READ(2,2002,END=22,ERR=22)FDFNAM
      IF(CKNAMF(FDFNAM,NAMFDF,3))GO TO 30
      CLOSEFILE 2
   22 CALL BREAK(1)
      PRINT     1004,FDFNAM
      GO TO 2
C     PRINT     1006
 1006 FORMAT(3X,16HEND-OF-TOPO-DATA)
C          READ FUEL TYPE
   30 READ(2,2005,END=22,ERR=22)IFTYPE
 2005 FORMAT(I5)
C          READ DATA HEADER
      READ(2,2003,END=22,ERR=22)KUADF,X0F,Y0F,XMF,YMF,DXFI,
     &              DXFII,DYFI,DYFII,DXF02,DYF02,NXFC,NYFC,NCELF
 2003 FORMAT(I5,4E15.7/4E15.7/ 2E15.7,2I5,I10)
C          READ AGE TABLE
      READ(2,2004,END=22,ERR=22)AGET
      CLOSEFILE 2
      CALL CLOSER(XDUM)
C     PRINT     1007
 1007 FORMAT(3X,16HEND-OF-FUEL-DATA)
C          ASSUME MET DATA HAS SAME BOUNDARIES AS FUEL DATA
      KUADW=KUADF
      X0W=X0F
      Y0W=Y0F
      XMW=XMF
      YMW=YMF
      DXWI=(XMW-X0W)/6.0
      DYWI=DXWI
      DXWII=1.0/DXWI
      DYWII=DXWII
      DXW02=DXWI*0.5
      DYW02=DXW02
      NXWP=7
      NYWP=7
      NPTSW=49
      CMESH=DXWI
      IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
      CALL BREAK(1)
C
C     INPUT FROM TERMINAL - RESOURCE FILE NAME AND TYPE
C
   40 PRINT     1008
 1008 FORMAT(3X,46H3. ENTER RESOURCE FILE NAME AND TYPE (OLD/NEW))
      PRINT     1003
      READ      2015,RDF,RFLTYP
 2006 FORMAT(A8,A4)
C          TYPE OF RESOURCE FILE
      RDFNAM(1)=TDFNAM(1)
      RDFNAM(2)=TDFNAM(2)
      RDFNAM(3)=TDFNAM(3)
      IF(RDF.EQ.TDF)GO TO 65
      RDFNAM(1)=FDFNAM(1)
      RDFNAM(2)=FDFNAM(2)
      RDFNAM(3)=FDFNAM(3)
      IF(RDF.EQ.FDF)GO TO 65
      RDFNAM(1)=BLANK
      RDFNAM(2)=BLANK
      RDFNAM(2)=BLANK
      IF(RFLTYP.EQ.OLD)GO TO 60
```

```
      IF(RFLTYP.EQ.NEW)GO TO 80
C          ERROR-FILE TYPE INVALID
   50 PRINT     1009
 1009 FORMAT(3X,24H***ERROR*** INVALID TYPE)
      GO TO 40
C
C     INPUT FROM RESOURCE DATA FILE - FILE TYPE
C
   60 CALL BREAK(0)
      READ(3,2002,END=65,ERR=65)RDFNAM
      IF(CKNAMF(RDFNAM,NAMRDF,3))GO TO 70
      CLOSEFILE 3
   65 CALL BREAK(1)
      PRINT     1004,RDFNAM
      GO TO 40
C
C     INPUT FROM RDF - RESOURCE DATA
C
   70 READ(3,2007,END=65,ERR=65)NRSTOT,NRSMI
      IF(NRSMI.NE.NRSMAX)GO TO 65
      READ(3,2008,END=65,ERR=65)IORD,IDR,NORS,ETA,ACT
C          REPOSITION FILE
      CLOSEFILE 3
      CALL CLOSER(XNUM)
      IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
      CALL BREAK(1)
      GO TO 90
C
C     CREATE NEW RESOURCE DATA FILE
C
   80 NRSTOT = 0
      DO 82 IZ=1,NRSMAX
      IORD(IZ)=IZ
      IDR(IZ)=0
      NORS(IZ)=0
      ETA(IZ)=0.0
      ACT(IZ)=0.0
   82 CONTINUE
      CALL BREAK(0)
      CALL CREATE(RDF," ",0,ISTAT)
      IF(ISTAT.EQ.0)GO TO 88
      CALL BREAK(1)
   85 PRINT     1010
 1010 FORMAT(3X,52H***ERROR*** FILE ALREADY EXISTS OR CANNOT BE CREATED)
      GO TO 40
C          WRITE LABEL
   88 RDFNAM(1)=NAMRDF(1)
      RDFNAM(2)=NAMRDF(2)
      RDFNAM(3)=NAMRDF(3)
      WRITE(3,2002)RDFNAM
      WRITE(3,2007)NRSTOT,NRSMAX
      WRITE(3,2008)IORD,IDR,NORS,ETA,ACT
      WRITE(3,2009)
 2009 FORMAT(20HEND-OF-RESOURCE-DATA)
      CLOSEFILE 3
      CALL CLOSER(XDUM)
      IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
      CALL BREAK(1)
```

```
C
C     INPUT APPROXIMATE FIRE SIZE
C
   90 PRINT     1033
 1033 FORMAT(3X,56H4. ENTER APPROXIMATE FIRE SIZE IN ACRES (0.0 IF UNKNOWN)
     &       )
      PRINT     1003
      READ      2015,AFIRSZ
 2018 FORMAT(F10.0)
C         CHANGE UNITS
      AREA1=AFIRSZ*4046.724
      IF(AREA1.LT.729.658)AREA1=729.658
C
C     INPUT INITIAL DISTANCE OF ROAD
C
      IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
   92 PRINT     1046
 1046 FORMAT(3X,51H5. ENTER INITIAL DISTANCE OFF ROAD (0.0 IF UNKNOWN)
     & 8H IN FEET)
      PRINT     1003
      READ      2015,DISTRI
      DISTRI=DISTRI*0.3048
      IF(DISTRI.LT.0.06096)DISTRI=60.96
C
C     INPUT FIRE TIME AND DATE
C
      IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
  172 PRINT     1044
 1044 FORMAT(3X,37H6. ENTER TIME OF FIRE (24 HOUR)        )
      PRINT     1003
      READ      2015,IFIRET
 2022 FORMAT(I4)
      CALL VERTIM(IFIRET,IFHRS,IFMIN,IFMFM,OK)
      TOF=FLOAT(IFMFM)
      IF(OK)GO TO 174
      PRINT     1016
      GO TO 172
C
  174 PRINT     1045
 1045 FORMAT(3X,27HENTER DATE OF FIRE (MMDDYY))
      PRINT     1003
      READ      2015,IDTFI
      CALL VERDAT(IDTFI,IDFI,IMFI,IYFI,JFI,OK)
      JDOF=JFI
      IF(OK)GO TO 91
      PRINT     1014
      GO TO 174
   91 IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
      RETURN
 1014 FORMAT(3X,24H***ERROR*** ILLEGAL DATE)
 1016 FORMAT(3X,24H***ERROR*** ILLEGAL TIME)
 2011 FORMAT(I6)
 2015 FORMAT(V)
 2008 FORMAT(6(10I5/),7(5E15.7/),5E15.7)
 2007 FORMAT(2I10)
      END
```

```
$$$$$SMP2
      $ LINK UMP2
      SUBROUTINE MP2(TDF,FDF,RDF,WATERA,RDFNAM)
C
C     EXPERIMENTAL INITIAL ATTACK EVALUATION PROGRAM (TO BE USED IN
C          EVALUATING THE POSSIBLE SUCCESS OR FAILURE OF AN INITIAL
C          ATTACK ON WILDLAND FIRES)
C     VER 3.00 (FOR USE ON G. E. TIMESHARE ONLY)
C
C     PROGRAMMER - JOHN M. SUNDERSON, JR.
C
      FILENAME RDF,FDF,TDF
C          FILES ARE AS FOLLOWS.......
C     TDF     - TOPO DATA FILE (PREPARED BY A SEPARATE PROGRAM)
C     FDF     - FUEL DATA FILE (PREPARED BY A SEPARATE PROGRAM)
C     RDF     - RESOURCES DATA FILE (CREATED AND ALTERED BY THIS PROGRAM)
C
C          ROUTINES USED
C     CKNAMF
C
      ALPHA WANS,GEOS,SURF
      ALPHA RDFNAM(3),NAMRDF(3),YES,NO,WATERA,ANS
      ALPHA TDFNAM(3),NAMTDF(3),FDFNAM(3),NAMFDF(3),RFLTYP,OLD,NEW
C
      LOGICAL OK
      LOGICAL CKNAMF,NOTOK
C
      COMMON/TOPO2/KUADT,XOT,YOT,XMT,YMT,DXTI,DXTII,DYTI,DYTII,SRUF,
     &                DXTO2,DYTO2,NXTP,NYTP,NPTST,HBC(51,51)
      COMMON/FUEL2/KUADF,XOF,YOF,XMF,YMF,DXFI,DXFII,DYFI,DYFII,
     &                DXFO2,DYFO2,NXFC,NYFC,NCELF,AGET(50,50),IFTYPE
C
      COMMON/ROUTI2/NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
     &                 ETA(20),ACT(20)
      COMMON/SPLMET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
      COMMON/MET3/WANS,T850C,DPH50C,WSMPM8,WDR8,
     &               KUADW,XOW,YOW,XMW,YMW,DXWI,DXWII,DYWI,DYWII,
     &               DXWO2,DYWO2,NXWP,NYWP,NPTSW,WINDSG(7,7),WINDDG(7,7),
     &               DVSURR,WVSURM,RHSUR,TSURC
      COMMON/FIRE1/XF,YF,AREA1,FTIN(11),NPTRQ,PTCAL(11),NPTCAL,IDISP
      COMMON/FMOD1/CMESH,TSPRED,ISPREP,METIN,CTHRSH
      COMMON/TIME1/JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
      COMMON/PERIM1/NPT,NXY,DELT,PT(11),PX(20,11),PY(20,11)
      COMMON/CNSTNT/RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
      COMMON /IMPFER/ TINCR,NTSTEP,PL(51),WDN(51),PCNR(51),PCNT(51),
     &                     PGR(51),CWR(51)
      COMMON/RSDATA/DECODI(2,8),MAXR,MAXS,MAXC,RESCHR(15,23),AUX(8)
      COMMON/MCON/IYFI,JFI
      COMMON/BREKIT/NSTEP,IQSTEP,IMSTEP,IPSTEP(12),
     & ICFLAG(12),IMMSTP(12)
C
      DATA GEOS/4HTEST/,SURF/4HSURF/
      DATA NAMRDF/4HRESO,4HURCE,4H-DAT/,YES/3HYES/,NO/2HNO/
      DATA OLD/3HOLD/,NEW/3HNEW/
      DATA NAMTDF/4HTOPO,4H-DAT,1HA/,NAMFDF/4HFUEL,4H-DAT,1HA/
C          OUTPUT CURENT RESOURCES
      JUMPS=IQSTEP-7
      GO TO (8,94,96,98,190),JUMPS
    8 CALL BREAK(0)
      WRITE(3,2002)RDFNAM
 2002 FORMAT(3A4)
      WRITE(3,2007)NRSTOT,NRSMAX
      WRITE(3,2008)IORD,IDR,NORS,ETA,ACT
```

```
C      NOTE...... MAY WANT TO CHANGE ORDER OF OUTPUT AND INPUT TO
C       REFLECT ETA ORDERING PER IORD
        WRITE(3,2009)
  2007 FORMAT(2I10)
  2008 FORMAT(6(10I5/),7(5E15.7/),5E15.7)
  2009 FORMAT(20HEND-OF-RESOURCE-DATA)
        CLOSEFILE 3
        CALL BREAK(I)
C
C       DETERMINE WATER AVAILABILITY
C
    92 PRINT    1011
  1011 FORMAT(3X,36H8. IS WATER AVAILABLE AT SITE (YES/NO))
        PRINT    1003
        READ    2010,WATERA
  2010 FORMAT(A4)
        IF(WATERA.EQ.YES)GO TO 93
        IF(WATERA.EQ.NO)GO TO 93
        PRINT    1012
  1012 FORMAT(3X,32H***ERROR*** UNRECOGNIZABLE ENTRY)
        GO TO 92
C
C       DETERMINE DATE OF LAST RAINFALL AND MET DATA TIME AND DATE
C
    93 IPSTEP(IQSTEP)=1
        IQSTEP=IQSTEP+1
C
    94 PRINT    1013
  1013 FORMAT(3X,45H9. ENTER DATE OF LAST RAINFALL (0 IF UNKNOWN)/
      & 3X,9HFORMAT IS/12X,6HMMDDYY)
        PRINT    1003
        READ    2011,IDLRI
        IF(IDLRI.LT.1)IDLRI=41500+IYFI
        CALL VERDAT(IDLRI,IDLR,IMLR,IYLR,JDLRF,OK)
        IF(OK)GO TO 95
        PRINT    1014
  1014 FORMAT(3X,24H***ERROR*** ILLEGAL DATE)
        GO TO 94
C          GET TOTAL DEPTH OF RAINFALL (DEPLR OR DRAIN)
    95 PRINT    1046
  1046 FORMAT(3X,42HENTER TOTAL RAINFALL LAST SEASON IN INCHES,
      & 15H (0 IF UNKNOWN))
        PRINT    1003
        READ    2015,DEPLR
        IF(DEPLR.LT.0.0001)DEPLR=12.5
        DRAIN=0.0254*DEPLR
        IF(DRAIN.GE.0.0.AND.DRAIN.LT.2.0)GO TO 396
        PRINT    1047
  1047 FORMAT(3X,30H***ERROR*** UNREASONABLE DEPTH)
        GO TO 95
   396 IPSTEP(IQSTEP)=1
        IQSTEP=IQSTEP+1
C          GET TIME OF MET DATA
  2011 FORMAT(I6)
    96 PRINT    1015
  1015 FORMAT(3X,38H10. ENTER TIME OF MET.DATA COLLECTION
      & /3X,9HFORMAT IS / 12X,4HHHMM)
        PRINT    1003
        READ    2022,METTIM
        CALL VERTIM(METTIM,MHRS, MMIN,MMFM,OK)
        TDOM=FLOAT(MMFM)$IF(OK)GO TO 97
        PRINT    1016
  1016 FORMAT(3X,24H***ERROR*** ILLEGAL TIME)
        GO TO 96
```

```
C            GET DATE OF MET  DATA
   97 PRINT     1043
 1043 FORMAT(3X,34HENTER DATE OF MET. DATA COLLECTION
     & /3X,9HFORMAT IS/12X,4HMMDD)
      PRINT     1003
      READ      2015,METDAT
      METDAT=METDAT*100+IYFI
      CALL VERDAT(METDAT,IDMD,IMMD,IYRM,IDOM,OK)
      IF(OK)GO TO 398
      PRINT     1014
      GO TO 97
  398 IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
C
C     FIND OUT WHAT TYPE MET DATA IS TO BE INPUT
C
   98 PRINT     1017
 1017 FORMAT(3X,67H1. ENTER ALTITUDE CLASSIFICATION OF MET DATA (GEOSTR
     &OPHIC/SURFACE))
      PRINT     1003
      READ      2013,WANS
 2012 FORMAT(I6)
 2013 FORMAT(A4)
      IF(WANS.EQ.GEOS)GO TO 100
      IF(WANS.EQ.SURF)GO TO 130
      PRINT     1018
 1018 FORMAT(3X,32H***ERROR*** UNRECOGNIZABLE INPUT)
      GO TO 98
C
C     INPUT FROM TERMINAL - GEOSTROPHIC MET DATA
C
C            GET GEOSTROPHIC TEMP.
  100 PRINT     1019
 1019 FORMAT(3X,38HENTER 850 MB  TEMPERATURE IN DEGREES F)
      PRINT     1003
      READ      2015,T850F
      IF(0.0.LE.T850F.AND.T850F.LE.130.0)GO TO 110
      PRINT     1020
 1020 FORMAT(3X,36H***ERROR*** UNREASONABLE TEMPERATURE/
     &          15X,            39HSHOULD BE BETWEEN 0.0 AND 130.0 DEGREES)
      GO TO 100
C            GET GEOSTROPHIC DEW POINT TEMPERATURE
  110 PRINT     1021
 1021 FORMAT(3X,46HENTER 850 MB  DEW POINT TEMPERATURE IN DEGREES F)
      PRINT     1003
      READ      2015,DP850F
 2014 FORMAT(F10.0)
      IF(-22.0.LE.DP850F.AND.DP850F.LE.130.0)GO TO 120
      PRINT     1022
 1022 FORMAT(3X,36H***ERROR*** UNREASONABLE TEMPERATURE/
     &          15X,41HSHOULD BE BETWEEN -22.0 AND 130.0 DEGREES)
      GO TO 110
C            GET WIND SPEED AND DIRECTION
  120 PRINT     1023
 1023 FORMAT(3X,76HENTER 850 MB WIND SPEED IN MPH AND DIRECTION IN DEGREE
     &S CLOCKWISE FROM NORTH)
      PRINT     1003
      READ      2015,WSMPH8,WDD8F
 2015 FORMAT(V)
      IF((0.0.LE.WSMPH8.AND.WSMPH8.LE.100.0).AND.
     &   (-360.0.LE.WDD8F.AND.WDD8F.LE.360.0))GO TO 122
      PRINT     1024
 1024 FORMAT(3X,48H***ERROR*** UNREASONABLE WIND SPEED OR DIRECTION/
     &          15X,42HWIND SPEED SHOULD BE BETWEEN 0 AND 100 MPH/
     &          15X,36HDIRECTION BETWEEN -360 AND 360 DEGREES)
      GO TO 120
```

```
C
C      PERFORM UNITS TRANSFORMATIONS FOR GEOSTROPHIC DATA
C
  122 T850C=0.55555556*(T850F-32.0)
      DP850C=0.55555556*(DP850F-32.0)
      WSMPM8=26.82167*WSMPH8
      WDR8=0.01745329*WDD8F
C
C      COMPUTE AN APPROXIMATE AVERAGE FOR GEOSTROPHIC DATA
C
      XC=X0W+FLOAT(NXWP-1)*DXWI*0.5
      YC=Y0W+FLOAT(NYWP-1)*DYWI*0.5
      CALL SWIND(WSMPM8,WDR8,TDOM,TOF,HEIGHT(XF,YF),1.0,AVEWS,AVEWD)
C
      GO TO 170
C
C      INPUT FROM TERMINAL - SURFACE MET DATA
C
C            GET SURFACE TEMPERATURE
  130 PRINT     1025
 1025 FORMAT(3X,38HENTER SURFACE TEMPERATURE IN DEGREES F)
      PRINT     1003
      READ      2015,TSURF
      IF(10.0.LE.TSURF.AND.TSURF.LE.130.0)GO TO 140
      PRINT     1026
 1026 FORMAT(3X,36H***ERROR*** UNREASONABLE TEMPERATURE/
     &        15X,42HSHOULD BE BETWEEN 10.0 AND 130.0 DEGREES F)
      GO TO 130
C            GET SURFACE REL. HUMIDITY
  140 PRINT     1027
 1027 FORMAT(3X,42HENTER SURFACE RELATIVE HUMIDITY IN PERCENT)
      PRINT     1003
      READ      2015,RHSUR
      RHSUR=RHSUR*0.01
 2016 FORMAT(F10.0)
 2017 FORMAT(F10.0)
      IF(0.0.LT.RHSUR.AND.RHSUR.LT.1.0)GO TO 150
      PRINT     1028
 1028 FORMAT(3X,42H***ERROR*** UNREASONABLE RELATIVE HUMIDITY/
     &        15X,25HSHOULD BE BETWEEN 0 AND 1)
      GO TO 140
C            GET SURFACE WIND SPEED AND DIRECTION AND VARIATIONS OF EACH
  150 PRINT     1029
 1029 FORMAT(3X,52HENTER SURFACE WIND SPEED IN MPH AND VARIATION IN MPH)
      PRINT     1003
      READ      2015,WSSURH,WVSURH
      IF((0.0.LE.WSSURH.AND.WSSURH.LE.100.0).AND.
     &   (0.0.LE.WVSURH.AND.WVSURH.LE.50.0))GO TO 152
      PRINT     1030
 1030 FORMAT(3X,48H***ERROR*** UNREASONABLE WIND SPEED OR VARIATION/
     &        15X,31HWINDSPEED BETWEEN 0 AND 100 MPH/
     &        15X,30HVARIATION BETWEEN 0 AND 50 MPH)
      GO TO 150
  152 PRINT     1031
 1031 FORMAT(3X,53HENTER SURFACE WIND DIRECTION IN DEGREES AND VARIATION
     & ,22H IN DEGREES FROM NORTH)
      PRINT     1003
      READ      2015,DWSURD,DVSURD
      DVSURD=ABS(DVSURD)
      IF((ABS(DWSURD).LE.360.0).AND.
     &   (     DVSURD .LE.180.0))GO TO 154
      PRINT     1032
 1032 FORMAT(3X,52H***ERROR*** UNREASONABLE WIND DIRECTION OR VARIATION/
     &        15X,43HWIND DIRECTION BETWEEN -360 AND 360 DEGREES/
     &        15X,38HVARIATION BETWEEN   0  AND 180 DEGREES)
      GO TO 152
```

```
C
C       CHANGE UNITS OF ALL SURFACE DATA
C
  154   TSURC=0.55555556*(TSURF-32.0)
        WSSURM=26.8216667*WSSURH
        WVSURM=26.8216667*WVSURH
        IF(DWSURD.LT.0.0)DWSURD=360.0+DWSURD
        DWSURR=0.01745329252*DWSURD
        DVSURR=0.01745329252*DVSURD
C
C       LOOP OVER ALL WIND POINTS TO GENERATE DATA AND CALC. AVERAGES
C
        AVEWS=0.0
        AVEWD=0.0
        DO 168 I=1,7
        DO 166 J=1,7
C
C 160   GENERATE DATA AT EACH POINT AND SUM FOR AVERAGES
C
        WS=RANGAU(WSSURM,WVSURM)
        WD=RANGAU(DWSURR,DVSURR)
C
        AVEWS=AVEWS+WS
        AVEWD=AVEWD+WD
C
        WINDSG(I,J)=WS
        WINDDG(I,J)=WD
C
  166   CONTINUE
  168   CONTINUE
        AVEWS=AVEWS/49.0
        AVEWD=AVEWD/49.0
C
C       BEGIN CALCULATION OF FIRE LOCATION
C
  170   IF(AVEWS.GE.134.10633)GO TO 180
C             USE CENTER OF DATA BASE
        XF=(XMW+XOW)*0.5
        YF=(YMW+YOW)*0.5
        GO TO 190
C             CALCULATE LOCATION IN DATA BASE
  180   FMAG=(XMW-XOW)*0.5
        XF=FMAG-0.6*FMAG*SIN(AVEWD)
        YF=FMAG-0.6*FMAG*COS(AVEWD)
        IPSTEP(IQSTEP)=1
        IQSTEP=IQSTEP+1
C
C       BEGIN INPUT OF EXISTING MODIFIDED FUEL LINES
C
C             ARE THERE ANY TO BE MADE
  190   PRINT     1034
 1034   FORMAT(3X,41HARE THERE ANY FUEL MODIFICATIONS (YES/NO))
        PRINT     1003
        READ      2019,ANS
 2019   FORMAT(A4)
        IF(ANS.EQ.NO)GO TO 220
        XFOUT=XF*3.28084
        YFOUT=YF*3.28084
        IF(ANS.EQ.YES)PRINT    1050,XFOUT,YFOUT
 1050   FORMAT(3X,31HFIRE IGNITION COORDINATES, X = ,F7.0,
     &  6H, Y = ,F7.0)
        IF(ANS.EQ.YES)GO TO 200
        PRINT     1035
 1035   FORMAT(3X,32H***ERROR*** ENTER ONLY YES OR NO)
```

```
          GO TO 190
C
C         INPUT FUEL MODIFICATION LINE
C
     200  PRINT     1036
    1036  FORMAT(3X,29HENTER STARTING POINT FOR LINE,
        & 28H TO BE MODIFIED(X,Y) IN FEET  )
          PRINT     1003
          READ    2015,XLM1,YLM1
          XLM1=XLM1*0.3048+XF
          YLM1=YLM1*0.3048+YF
          IF(XOF.LE.XLM1 .AND. XLM1.LE.XMF
         &                    .AND.
         &    YOF.LE.YLM1 .AND. YLM1.LE.YMF)GO TO 202
          PRINT     1037
    1037  FORMAT(3X,54H***ERROR*** FUEL MODIFICATION LINE OUTSIDE FUEL BOUND
         &S)
          GO TO 200
     202  PRINT     1038
    1038  FORMAT(3X,35HENTER DIRECTION OF LINE IN DEGREES ,
        & 20HCLOCKWISE FROM NORTH)
          PRINT     1003
          READ     2015,DIRLD
          IF(-360.0.LE.DIRLD.AND.DIRLD.LE.360.0)GO TO 204
          PRINT     1039
    1039  FORMAT(3X,30H***ERROR*** DIRECTION IN ERROR)
    2020  FORMAT(F10.0)
          GO TO 202
     204  DIRLR=DIRLD*0.01745329252
          IF(DIRLR.LT.0.0)DIRLR=6.2831853+DIRLR
     206  PRINT     1040
    1040  FORMAT(3X,46HENTER LENGTH OF FUEL MODIFICATION LINE IN FEET)
          PRINT     1003
          READ     2015,RLLEN
          RLLEN=ABS(RLLEN)*0.3048
          XLM2=XLM1+COS(1.5707963268-DIRLR)*RLLEN
          YLM2=YLM1+SIN(HALFPI-DIRLR)*RLLEN
          IF(XOF.LE.XLM2.AND.XLM2.LE.XMF
         &                   .AND.
         &    YOF.LE.YLM2.AND.YLM2.LE.YMF)GO TO 210
          PRINT     1041
    1041  FORMAT(3X,43H***ERROR*** FUEL MODIFICATION LINE TOO LONG)
          GO TO 206
C
C         SET ALL AGE CELLS NEGATIVE THAT THIS LINE CROSSES
C
     210  CALL SETLN2(XLM1,YLM1,XLM2,YLM2,DXFI,XOF,YOF,NXFC,NYFC)
     212  CALL GETLN2(IXO,IYO,NOTOK)
          IF(NOTOK)GO TO 214
          AGET(IXO,IYO)=-ABS(AGET(IXO,IYO))
          GO TO 212
C
C         ARE THERE MORE FUEL MODIFICATION LINES
C
     214  PRINT     1042
    1042  FORMAT(3X,39HARE THERE MORE FUEL MOD. LINES (YES/NO))
          PRINT     1003
          READ     2021,ANS
    2021  FORMAT(A4)
          IF(ANS.EQ.YES)GO TO 200
          IF(ANS.EQ.NO)GO TO 220
          PRINT     1035
```

```
      GO TO 214
C
C     PREPARE INPUTS FOR PERIMETER CALCULATION
C
  220 CMESH=200.0
      TINCH=12.0
      NPTRQ=5
      NXY=4
      RETURN
 1003 FORMAT(7X,4H....)
 2022 FORMAT(I4)
      END
```

```
$$$$$SRS
      $ LINK URS
      SUBROUTINE RS
      COMMON/RSDATA/DAT(372)
      OPENFILE "RCDATA"("SYSTEM")
      READ("RCDATA")DAT
      CLOSEFILE "RCDATA"
      RETURN
      END
```

4.0 CHECK ALPHABETIC NAME STRINGS (CKNAMF)

1. Purpose

This function compares two alphabetic strings to determine if they are identical.

2. Arguments

INPUTS

A1	-	Alpha name string one
A2	-	Alpha name string two
LEN	-	Length of both strings

OUTPUTS

CKNAMF	-	Logical comparison result (True=identical)

3. Procedure

Each portion of each string is compared. If any comparison fails a CKNAMF is set to false.

4. Comments

None.

5. Flow Chart

Not required.

```
      LOGICAL FUNCTION CKNAMF(A1,A2,LEN)
C
C     FUNCTION TO COMPARE TWO ALPHA NAME STRINGS
C     VER 1.00     (G. E. TIMESHARE ONLY)
C
C     PROGRAMMER - JOHN M. SUNDERSON, JR.
C
      ALPHA A1(LEN),A2(LEN)
C
C         INPUTS
C     A1       - ALPHA NAME STRING ONE
C     A2       - ALPHA NAME STRING TWO
C     LEN      - LENGTH OF STRINGS
C         OUTPUTS
C     CKNAMF - COMPARISON RESULT (TRUE=SAME)
C
      CKNAMF = .TRUE.
C
      DO 10 I=1,LEN
      IF(A1(I).EQ.A2(I))GO TO 9
      CKNAMF = .FALSE.
      GO TO 15
    9 CONTINUE
   10 CONTINUE
   15 CONTINUE
      RETURN
      END
```

5.0 VERIFY DATE (VERDAT)

1. Purpose

This routine converts (and verifies) a packed integer of the form MMDDYY to day, month, year and days from December 31.

2. Arguments

<u>INPUT</u>

IMDY	-	Date in 6 digit packed integer (MMDDYY)

<u>OUTPUTS</u>

IDAY	-	Day of month
IMON	-	Month of year
IYR	-	Year (years-1900)
IJULN	-	Days from December 31
OK	-	Validity of result (TRUE=valid)

3. Procedure

The day, the month, and year are unpacked by multiplication and division. The days from December 31 are calculated from a table giving the total days at the end of each month. Validity checks are placed as required (e.g., no month may be greater than 12).

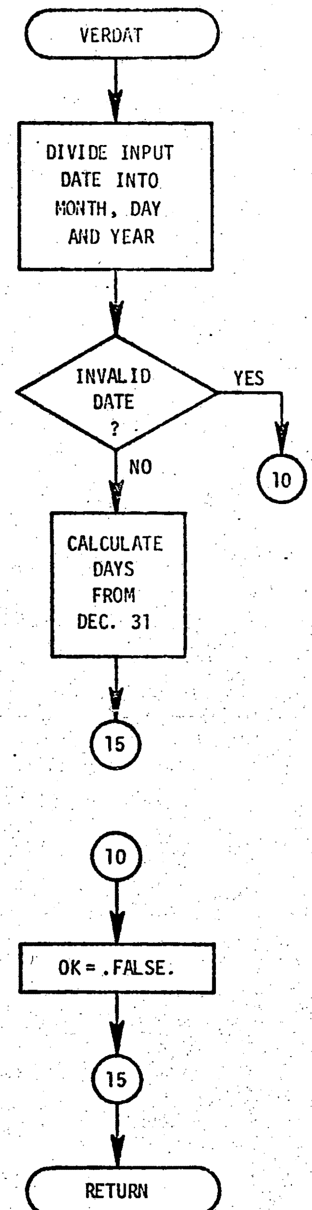
4. Comments

None.

5. Flow Chart

See Figure 2.

Figure 2 Verify Date



```
      SUBROUTINE VERDAT(IMDY,IDAY,IMON,IYR,IJULN,OK)
C
C     THIS ROUTINE CONVERTS (AND VERIFIES) A PACKED INTEGER OF THE
C     FORM MMDDYY TO DAY, MONTH, YEAR, AND DAYS FROM DEC 31(JULIAN)
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUT
C     IMDY    - DATE IN 6 DIGIT PACKED INTEGER(MMDDYY)
C         OUTPUTS
C     IDAY    - DAY OF MONTH
C     IMON    - MONTH OF YEAR
C     IYR     - YEAR (YEARS-1900)
C     IJULN   - DAYS GROM DEC 31 (JULIAN DATE WITH NO YEAR)
C     OK      - VALIDITY OF RESULT (TRUE=VALID)
C
      LOGICAL OK
C
      DIMENSION ISUMD(12)
      DIMENSION IDMTAB(12)
C
C                    JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC
      DATA IDMTAB/ 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31/
      DATA ISUMD/0,31,59,90,120,151,181,212,243,273,304,334/
C
      OK=.TRUE.
      IF(IMDY.LE.0)GO TO 10
      IMON=IMDY/10000
      ICHOP=IMON*10000
      IDAY=(IMDY-ICHOP)/100
      IYR=IMDY-ICHOP-IDAY*100
      IF(IDAY.LE.0)GO TO 10
      IF(IMON.LE.0)GO TO 10
      IF(IMON.GT.12)GO TO 10
      IDMTAB(2)=28
      IF(MOD(IYR,4).EQ.0)IDMTAB(2)=29
      IF(IDAY.GT.IDMTAB(IMON))GO TO 10
      IJULN=ISUMD(IMON)+IDAY
      IF(IMON.GT.2.AND.MOD(IYR,4).EQ.0)IJULN=IJULN+1
      GO TO 15
   10 OK=.FALSE.
   15 RETURN
      END
```

6.0 VERIFY TIME (VERTIM)

1. Purpose

This routine converts and verifies a packed integer of the form HHMM to hours and minutes.

2. Arguments

<u>INPUTS</u>

IHRMN	-	Time in four digit packed integer

<u>OUTPUTS</u>

IHRS	-	Hours
IMIN	-	Minutes
IMFM	-	Minutes from midnight
OK	-	Validity of result (TRUE=valid)

3. Procedure

The time is unpacked by division and checked for validity.

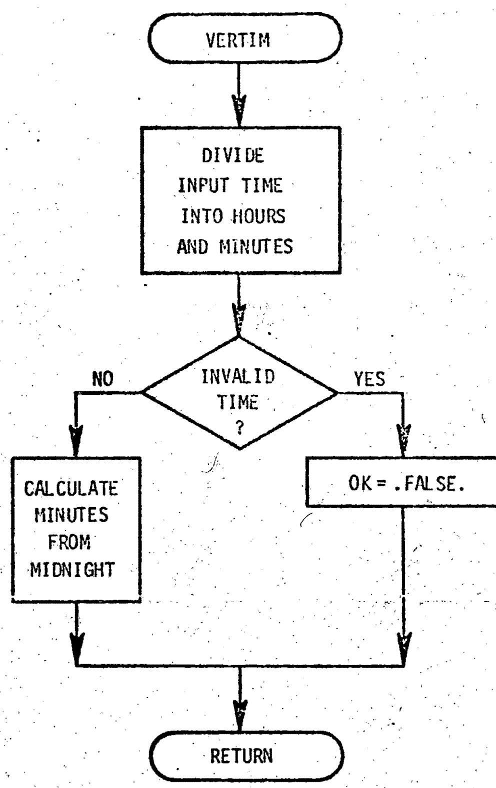
4. Comments

None.

5. Flow Chart

See Figure 3

Figure 3 Verify Time



```
      SUBROUTINE VERTIM(IHRMN,IHRS,IMIN,IMFM,OK)
C
C     THIS ROUTINE CONVERTS (AND VERIFIES )A PACKED INTEGER OF THE
C     FORM HHMM TO HOURS AND MINUTES
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUT
C     IHRMN  - TIME IN FOUR DIGIT PACKED INTEGER (24 HOUR TIME)
C         OUTPUTS
C     IHRS   - HOURS (FROM MIDNIGHT)
C     IMIN   - MINUTES (FROM THE HOUR)
C     IMFM   - MINUTES FROM MIDNIGHT
C     OK     - VALIDITY OF RESULT (TRUE=VALID)
C
      LOGICAL OK
C
      OK=.TRUE.
      IF(IHRMN.LT.0)OK=.FALSE.
      IHRS=IHRMN/100
      IMIN=IHRMN-IHRS*100
      IMFM=IHRS*60+IMIN
      IF(IHRS.GT.24.OR.IMIN.GT.59)OK=.FALSE.
C
      RETURN
      END
```

7.0 CONVERT DAY OF YEAR TO MONTH AND DAY OF MONTH (CNDYMD)

1. Purpose

This routine is used to convert the (Julian) day of the year to the corresponding month and day of the month.

2. Arguments

<u>INPUTS</u>

DAYOY	-	Day of year (days)
YEAR	-	Year (years-1900)

<u>OUTPUTS</u>

MONTH	-	Month corresponding to DAYOY
DAYOM	-	Day of month corresponding to DAYOY.

3. Procedure

The results are calculated using a loop which successively subtracts the days in each month and counts the months.

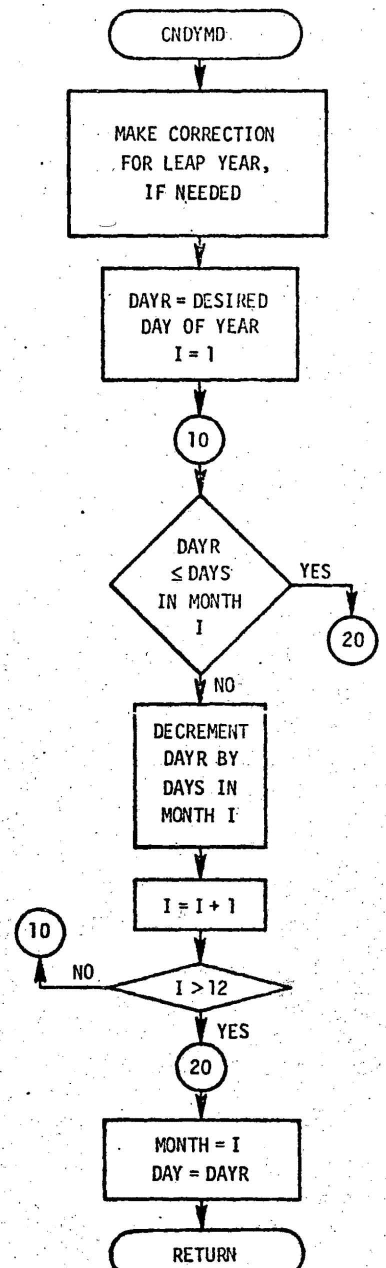
4. Comments

None.

5. Flow Chart

See Figure 4.

Figure 4 Convert Day of Year to Month and Day



```
      SUBROUTINE CNDYMD(DAYOY,YEAR,MONTH,DAYOM)
C
C     THIS ROUTINE CONVERTS DAY-OF-YEAR TO MONTH AND DAY-OF-MONTH
C     VER 1.00 (SIMILAR TO DAYCON)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     DAYOY  - DAY-OF-YEAR (INTEGER)
C     YEAR   - YEAR (INTEGER, YEAR-1900)
C         OUPUTS
C     MONTH  - MONTH-OF-YEAR (INTEGER, 1 TO 12)
C     DAYOM  - DAY-OF-MONTH (INTEGER)
C
      INTEGER DAYOY,YEAR,MONTH,DAYOM,DAYR,DINMON
C
      DIMENSION IDMTAB(12)
C                    JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC
      DATA IDMTAB/ 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31/
C
C         CHECK FOR LEAP YEAR
      IDMTAB(2)=28
      IF(MOD(YEAR,4).EQ.0)IDMTAB(2)=29
C         CALCULATE MONTH AND DAY
      DAYR=DAYOY
      DO 10 I=1,12
      DINMON=IDMTAB(I)
      K=I
      IF(DAYR.LE.DINMON)GO TO 20
      DAYR=DAYR-DINMON
   10 CONTINUE
   20 MONTH=K
      DAYOM=DAYR
C
      RETURN
      END
```

8.0 CONVERT RELATIVE TIME TO SIMULATED TIME (CNRTST)

1. Purpose

This routine converts relative time to simulated time.

2. Arguments

INPUTS

BTIM	-	Base time (minutes from midnight)
BDAY	-	Base day (days from December 31)
BYEAR	-	Base year (years-1900)
TINCR	-	Time increment (minutes from midnight)

OUTPUTS

NTIM	-	New simulated time (minutes from midnight)
NDAY	-	New day (days from December 31)
NYEAR	-	New year (years-1900)

3. Procedure

The time increment is added to the base time to give the new time, day and year. The new quantities are then adjusted such that time does not exceed 24 hours, and days do not exceed 365 (366 on leap year).

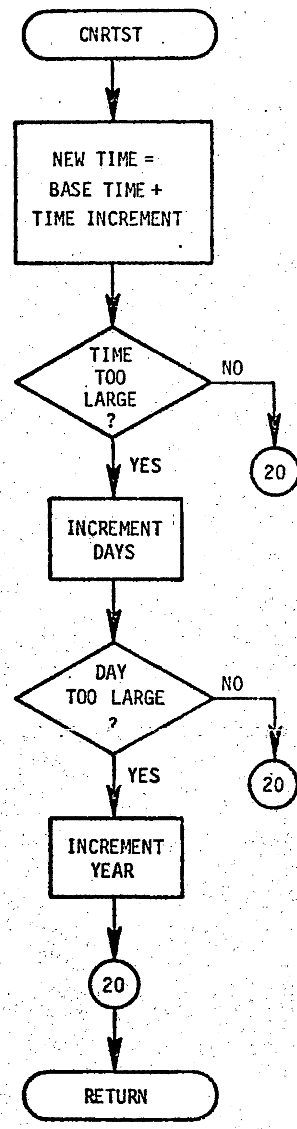
4. Comments

None.

5. Flow Chart

See Figure 5.

Figure 5 Relative-to-Simulated Time Conversion



```
      SUBROUTINE CNRTST(BTIM,BDAY,BYEAR,TINCR,NTIM,NDAY,
     & NYEAR)
C
C     THIS ROUTINE CONVERTS FROM RELATIVE TO SIMULATION TIME
C     VER 1.00 (SIMILAR TO STEPSM VER 1.10)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C          INPUTS FROM LIST
C     BTIM    - BASE TIME (REAL, MINUTES FROM MIDNIGHT)
C     BDAY    - BASE DAY (INTEGER, DAYS FROM DEC 31)
C     BYEAR   - BASE YEAR (INTEGER, YEARS-1900)
C     TINCR   - TIME INCREMENT (REAL, MINUTES FROM MIDNIGHT)(.LE.1YEAR)
C          OUTPUTS TO LIST
C     NTIM    - NEW TIME (REAL, MINUTES FROM MIDNIGHT)
C     NDAY    - NEW DAY (INTEGER, DAYS FROM DEC 31)
C     NYEAR   - NEW YEAR (INTEGER, YEARS-1900)
C
      REAL NTIM
      INTEGER BDAY,BYEAR
C
      NTIM=BTIM+TINCR
      NDAY=BDAY
      NYEAR=BYEAR
C
C          INCREMENT DAYS IF NECESSARY
      IF(NTIM.LT.1440.0)GO TO 20
      INCD=IFIX(NTIM/1440.0)
      NTIM=NTIM-FLOAT(INCD)*1440.0
      NDAY=NDAY+INCD
C          COMPUTE DAYS IN YEAR
      IDY=365
      IF(MOD(BYEAR,4).EQ.0)IDY=366
C          INCREMENT YEARS IF NECESSARY
      IF(NDAY.LE.IDY)GO TO 20
      NDAY=NDAY-IDY
      NYEAR=NYEAR+1
   20 RETURN
      END
```

9.0 GAUSSIAN RANDOM NUMBER CALCULATION (RANGAU)

1. Purpose

This function calculates a series of random numbers with Gaussian distribution.

2. Arguments

INPUTS

AM	-	Arithmetic mean
SDEV	-	Standard deviation

OUTPUTS

RANGAU	-	A random number with the normal distribution specified in the inputs.

3. Procedure

This routine calculates a normally distributed random number from a uniformly distributed random number series. The following approximation[2] is used:

$$Y = \frac{\sum_{i=1}^{k} X_i - \frac{K}{2}}{\sqrt{K/12}}$$

where

X_i is a uniformly distributed random number between 0 and 1 exclusive, and

K is the number of S_i's used.

In RANGAU, K is chosen as 12 to reduce calculations. The approximation then becomes:

$$Y = \sum_{i=1}^{12} X_i - 6$$

To achieve the required distribution the following adjustment is made:

$$Y' = Y \cdot S + AM$$

where

Y' is the normally distributed random number

S is the required standard deviation

AM is the required arithmetic mean

4. Comments

The distribution is clipped such that no value may be less than zero.

5. Flow Chart

Not required.

```
10010      FUNCTION RANGAU(AM,SDEV)
10020C
10030C     THIS ROUTINE CALCULATES A RANDOM NUMBER WITH GAUSSIAN (NORMAL)
10040C     DISTRIBUTION.
10050C     VER 1.00  (DESIGNED FOR USE ON G.E. TIMESHARE ONLY)
10060C
10070C     PROGRAMMER - JOHN SUNDERSON, JR.
10080C
10090C        INPUTS
10100C     AM     - ARITHMETIC MEAN OF REQUIRED RANDOM VARIABLE
10110C     SDEV   - STANDARD DEVIATION OF REQUIRED RANDOM VARIABLE
10120C        OUTPUT
10130C     RANGAU - A RANDOM NUMBER WITH NORMAL DISTRIBUTION
10140C        ROUTINES USED
10150C     RND-SYS
10160C
10170      SUM=RND(X)
10180      SUM=RND(X)+RND(X)+RND(X)+RND(X)+RND(X)+SUM
10190      SUM=RND(X)+RND(X)+RND(X)+RND(X)+RND(X)+RND(X)+SUM
10200      RANGAU=(SUM-6.0)*SDEV+AM
10205      IF(RANGAU.LT.0.0)RANGAU=0.0
10210      RETURN
10220      END
```

10.0 STORE LOCATION OF A LINE (SETLN2)

1. Purpose

This routine performs setup calculations for GETIN2 by storing grid and line characteristics.

2. Arguments

Ecah of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

The purpose of GETIN2 is to provide the cell indices along a line. SETLN2 first selects a direction in which to traverse the line segment by choosing the end of the segment with the smallest Y coordinate as the starting point. Next, the first and last cells on the line are calculated. Special cases (e.g., slope = 0.0) are removed and control is returned to the calling routine.

4. Comments

None.

5. Flow Chart

See Figure 6.

Figure 6 Line Location

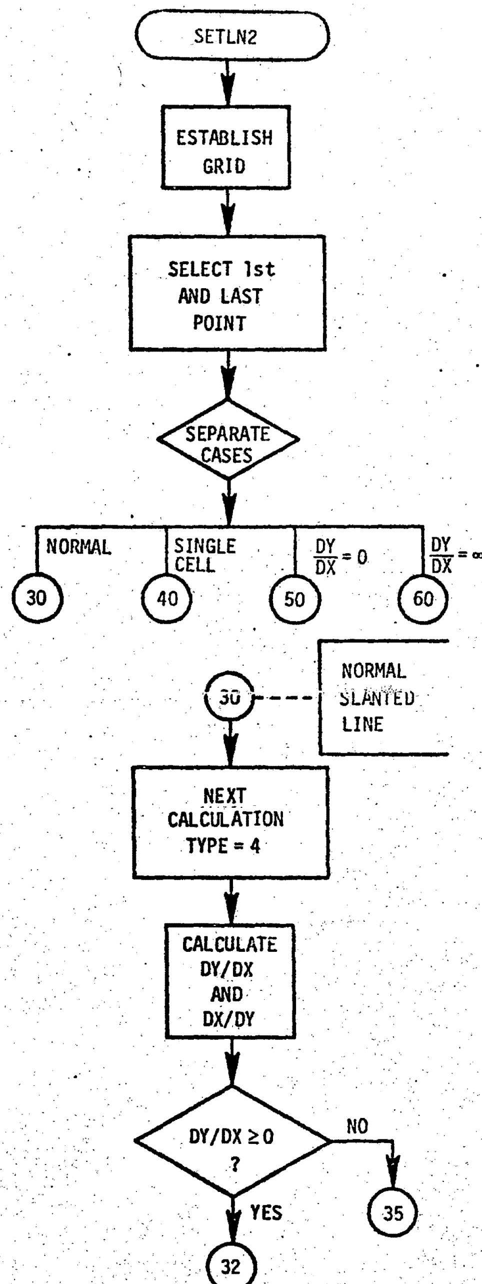


Figure 6 Line Location (Cont'd)

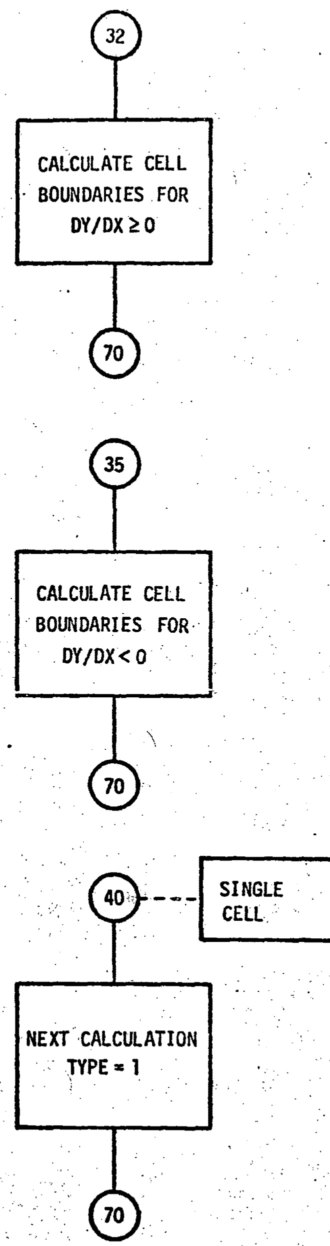
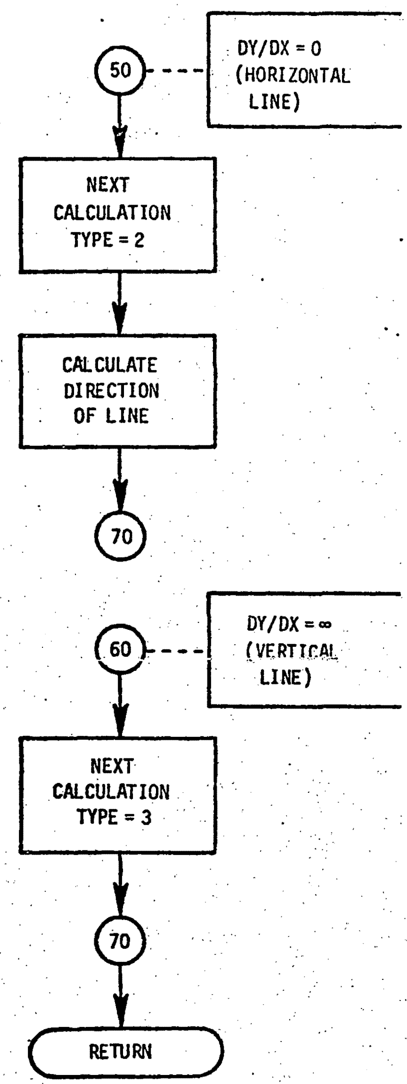


Figure 6 Line Location (Cont'd)



```
      SUBROUTINE SETLN2(X1,Y1,X2,Y2,GRDSZL,XMINGL,YMINGL,NXL,NYL)
C
C     THIS ROUTINE PERFORMS SETUP CALCULATIONS FOR GETIN2 STORING GRID
C     AND LINE CHARACTERISTICS
C     VER 1.00  (SIMILAR TO SETLIN VER 2.0)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C             INPUTS FROM LIST
C     X1       - X COORDINATE OF POINT 1
C     Y1       - Y COORDINATE OF POINT 1
C     X2       - X COORDINATE OF POINT 2
C     Y2       - Y COORDINATE OF POINT 2
C     GRDSZL - SIZE OF ONE GRID CELL (SQUARE CELLS)
C     XMINGL - MINIMUM X COORDINATE OF GRID
C     YMINGL - MINIMUM Y COORDINATE OF GRID
C     NXL      - NUMBER OF CELLS IN X DIRECTION
C     NYL      - NUMBER OF CELLS IN Y DIRECTION
C             OUTPUTS TO LINCL2 COMMON ASSOCIATED WITH LINE
C     X        - CURRENT X COORDINATE ON LINE
C     Y        - CURRENT Y COORDINATE ON LINE
C     IX       - CURRENT CELL INDEX X DIRECTION
C     IY       - CURRENT CELL INDEX Y DIRECTION
C     DYDX      - CHANGE IN Y WRT X
C     DXDY     - CHANGE IN X WRT Y
C     INC      - DIRECTION TO INCREMENT (+1 OR -1)
C     IXH      - LAST (HIGHEST CELL INDEX IN X DIRECTION
C     IYH      - LAST (HIGHEST) CELL INDEX IN Y DIRECTION
C     DXC      - INCREMENTAL CHANGE IN X DIRECTION (+ OR -)
C     BL       - LEFT BOUNDARY OF A CELL
C     BR       - RIGHT BOUNDARY OF A CELL
C     BU       - UPPER BOUNDARY OF A CELL
C     BB       - BOTTOM BOUNDARY OF A CELL
C     PHX      - LAST (HIGHEST) POINT ON LINE, X COORDINATE OF
C     PHY      - Y COORDINATE OF LAST (HIGHEST) POINT ON LINE
C     JUMP     - NEXT ACTION TO BE TAKEN BY GETIN2
C             OUTPUTS TO LINCL2 COMMON ASSOCIATED WITH GRID
C     XMING    - MINIMUM X COORDINATE OF GRID
C     YMING    - NIMIMUM Y COORDINATE OF GRID
C     XMAXG    - MAXIMUM X COORDINATE OF GRID
C     YMAXG    - MAXIMUM Y COORDINATE OF GRID
C     NX       - NUMBER OF CELLS IN X DIRECTION
C     NY       - NUMBER OF CELLS IN Y DIRECTION
C     GRDSIZ - SIZE OF ONE GRID CELL (SQUARE CELLS)
C     GRSIZI - GRDSIZ INVERSE
C
      COMMON/LINCL2/X,Y,IX,IY,DYDX,DXDY,INC,IXH,IYH,DXC,BL,BR,BU,BB,
     &                  PHX,PHY,JUMP,     XMING,YMING,XMAXG,YMAXG,NX,NY,
     &                  GRDSIZ,GRSIZI
C
C     ESTABLISH GRID DATA FROM LIST INPUT
C
      XMING=XMINGL
      YMING=YMINGL
      NX=NXL
      NY=NYL
      GRDSIZ=GRDSZL
      XMAXG=XMING+FLOAT(NX)*GRDSIZ
      YMAXG=YMING+FLOAT(NY)*GRDSIZ
      GRSIZI=1.0/GRDSIZ
      GRS02 =GRDSIZ*0.5
```

```
C
C      CALCULATE CELL INDICES OF END POINTS AND SELECT 1ST POINT
C
       IF(Y1.LE.Y2)GO TO 5
C         LOWEST POINT IS POINT 2
       PLX=X2
       PLY=Y2
       PHX=X1
       PHY=Y1
       GO TO 7
C         LOWEST POINT IS POINT 2
     5 PLX=X1
       PLY=Y1
       PHX=X2
       PHY=Y2
C         CALCULATE CELL INDICIES
     7 X=PLX
       Y=PLY
       IX=IFIX((X-XMING)*GRSIZI)+1
       IY=IFIX((Y-YMING)*GRSIZI)+1
       IXH=IFIX((PHX-XMING)*GRSIZI)+1
       IF(ABS((PHX-PLX)*GRSIZI).LE.0.01)IXH=IX
       IYH=IFIX((PHY-YMING)*GRSIZI)+1
       IF(ABS((PHY-PLY)*GRSIZI).LE.0.01)IYH=IY
       IX=MIN0(NX,MAX0(1,IX))
       IY=MIN0(NY,MAX0(1,IY))
       IXH=MIN0(NX,MAX0(1,IXH))
       IYH=MIN0(NY,MAX0(1,IYH))
C
C      REMOVE SPECIAL CASES
C
C         REMOVE SINGLE CELL CONTAINMENT
       IF(IX.EQ.IXH .AND. IY.EQ.IYH)GO TO 40
C         REMOVE DY/DX = 0
       IF(IY.EQ.IYH)GO TO 50
C         REMOVE DY/DX = INFINITY
       IF(IX.EQ.IXH)GO TO 60
C         NORMAL CASE
    30 JUMP=4
C
C      CALCULATE DY/DX AND DX/DY
C
       DELX=PHX-PLX
       DELY=PHY-PLY
       DYDX=DELY/DELX
       DXDY=DELX/DELY
C
C      CALCULATE CELL BOUNDARIES FOR DYDX.GT.0
C
       XC=FLOAT(IX-1)*GRDSIZ+XMING+GRS02
       YC=FLOAT(IY-1)*GRDSIZ+YMING+GRS02
       IF(DYDX.LT.0.0)GO TO 35
C
    32 BL=XC-GRS02
       BR=XC+GRS02
       BU=YC+GRS02
       BB=YC-GRS02
       INC=+1
       DXC=GRDSIZ
       GO TO 70
```

```
C
C     CALCULATE CELL BOUNDARIES FOR DYDX.LT.0
C
   35 BL=XC+GRSO2
      BR=XC-GRSO2
      BU=YC+GRSO2
      BB=YC-GRSO2
      INC=-1
      DXC=-GRDSIZ
      GO TO 70
C
C     SET PROPER ACTION FOR GETIN2
C
   40 JUMP=1
      GO TO 70
   50 JUMP = 2
      INC=+1
      IF(IX.LT.IXH)GO TO 70
      INC=-1
      GO TO 70
   60 JUMP = 3
C
   70 RETURN
      END
```

11.0 RETRIEVE CELL INDEX ON A LINE (GETIN2)

1. Purpose

This routine calculates each cell index along a line segment in a grided area.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

The coordinates of a point on the line segment of interest are maintained. This point is advanced along the line segment by calculating the coordinates of the intersection of the line with the cell boundaries of the current cell. As each new cell is encountered the index of that cell is calculated and returned to the calling routine.

4. Comment

None.

5. Flow Chart

See Figure 7.

Figure 7 Retrieve Cell Index

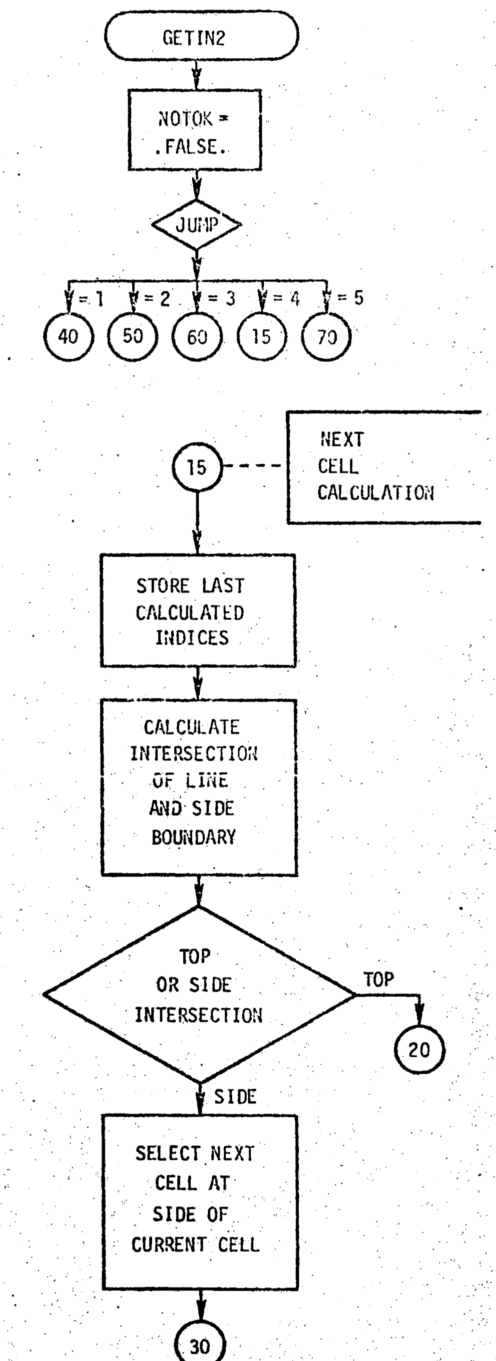


Figure 7 Retrieve Cell Index (Cont'd)

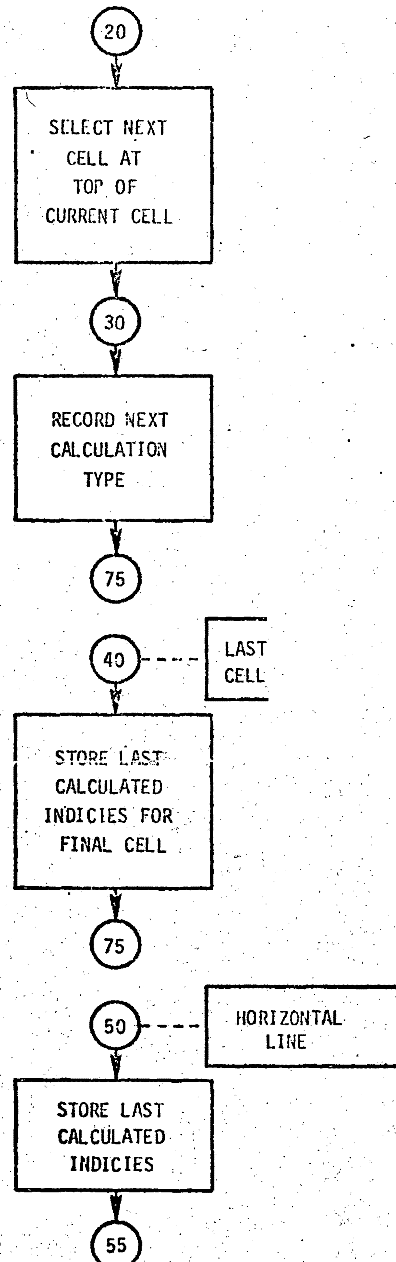
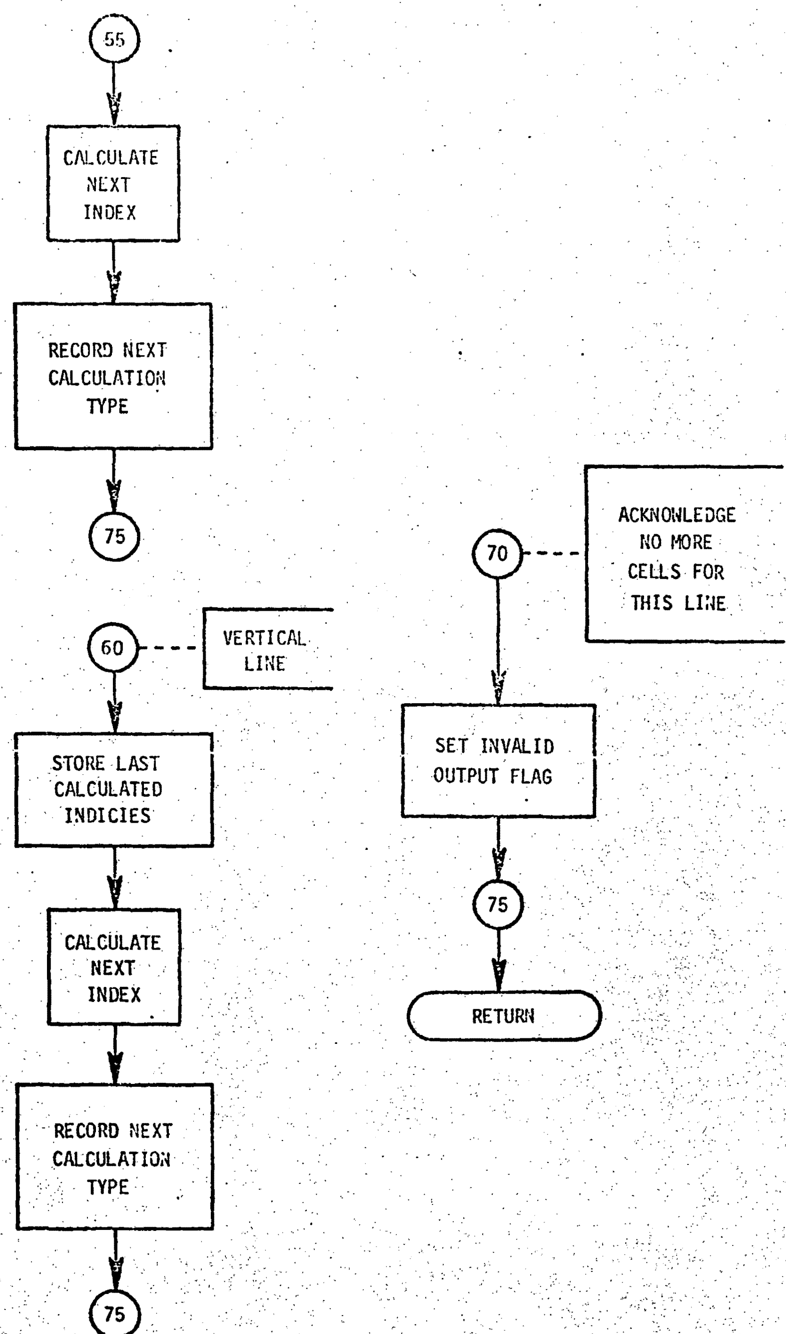


Figure 7 Retrieve Cell Index (Cont'd)



```
      SUBROUTINE GETIN2(IXO,IYO,NOTOK)
C
C     THIS ROUTINE RETURNS ALL CELL INDICIES ALONG A LINE DESCRIBED
C     BY ROUTINE SETLN2.  ONE CELL INDEX PAIR IS RETURNED PER CALL.
C     WHEN NO MORE CELLS ARE LEFT ON THE LINE, NOTOK IS SET TO .TRUE.,
C     OTHERWISE NOTOK IS .FALSE. .
C     VER 1.00 (SIMILAR TO GETIND)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS FROM LIST
C     NONE - ALL INPUTS ARE LOADED INTO COMMON /LINCL2/ BY SETLN2
C         INPUTS FROM COMMON /LINCL2/ ASSOCIATED WITH LINE
C     X        - CURRENT X COORDINATE ON LINE
C     Y        - CURRENT Y COORDINATE ON LINE
C     IX       - CURRENT CELL INDEX X DIRECTION
C     IY       - CURRENT CELL INDEX Y DIRECTION
C     DYDX      - CHANGE IN Y WRT X
C     DXDY     - CHANGE IN X WRT Y
C     INC      - DIRECTION TO INCREMENT (+1 OR -1)
C     IXH      - LAST (HIGHEST CELL INDEX IN X DIRECTION
C     IYH      - LAST (HIGHEST) CELL INDEX IN Y DIRECTION
C     DXC      - INCREMENTAL CHANGE IN X DIRECTION (+ OR -)
C     BL       - LEFT BOUNDARY OF A CELL
C     BR       - RIGHT BOUNDARY OF A CELL
C     BU       - UPPER BOUNDARY OF A CELL
C     BB       - BOTTOM BOUNDARY OF A CELL
C     PHX      - LAST (HIGHEST) POINT ON LINE, X COORDINATE OF
C     PHY      - Y COORDINATE OF LAST (HIGHEST) POINT ON LINE
C     JUMP     - NEXT ACTION TO BE TAKEN BY GETIN2
C         INPUTS FROM COMMON /LINCL2/ ASSOCIATED WITH GRID
C     XMING  - MINIMUM X COORDINATE OF GRID
C     YMING  - NIMIMUM Y COORDINATE OF GRID
C     XMAXG  - MAXIMUM X COORDINATE OF GRID
C     YMAXG  - MAXIMUM Y COORDINATE OF GRID
C     NX       - NUMBER OF CELLS IN X DIRECTION
C     NY       - NUMBER OF CELLS IN Y DIRECTION
C     GRDSIZ - SIZE OF ONE GRID CELL (SQUARE CELLS)
C     GRSIZI - GRDSIZ INVERSE
C         OUTPUTS TO LIST
C     IXO      - X INDEX OF CELL
C     IYO      - Y INDEX OF CELL
C     NOTOK    - RESULT VALIDITY FLAG
C
      COMMON/LINCL2/X,Y,IX,IY,DYDX,DXDY,INC,IXH,IYH,DXC,BL,BR,BU,BB,
     &                 PHX,PHY,JUMP,     XMING,YMING,XMAXG,YMAXG,NX,NY,
     &                 GRDSIZ,GRSIZI
C
      LOGICAL NOTOK
C
C
      NOTOK=.FALSE.
      GO TO (40,50,60,15,70),JUMP
   15 IXO=IX
      IYO=IY
C
C     CALCULATE INTERSECTION OF LINE AND RIGHT (OR LEFT) BOUNDARY
C
      YRB=DYDX*(BR-X)+Y
C
C     DOES LINE INTERSECT TOP OR SIDE OF CELL
```

```
C
      IF(YRB.GT.BU)GO TO 20
C
C     MOVE TO CELL ON RIGHT (OR LEFT) OF THIS CELL (LINE PENETRATES
C     CELL SIDE)
C
      X=BR
      Y=YRB
      IX=IX+INC
      BL=BR
      BR=BL+DXC
      GO TO 30
C
C     MOVE TO CELL ABOVE (LINE PENETRATES TOP)
C
   20 X=DXDY*(BU-Y)+X
      Y=BU
      IY=IY+1
      BB=BU
      BU=BB+GRDSIZ
C
C     HAVE ALL CELLS BEEN CALCULATED
C
   30 IF(IX.EQ.IXH .AND. IY.EQ.IYH)JUMP=1
      IF(Y.GT.PHY)JUMP=1
      GO TO 75
C
C     LAST CELL TO BE OUTPUT
C
   40 IXO=IX
      IYO=IY
C         NEXT ENTRY RETURNS NO CELL
      JUMP=5
      GO TO 75
C
C     HORIZONTAL LINE
C
   50 IXO=IX
      IYO=IY
      IX=IX+INC
      IF(IX.EQ.IXH)JUMP=1
      GO TO 75
C
C     VERTICAL LINE
C
   60 IXO=IX
      IYO=IY
      IY=IY+1
      IF(IY.EQ.IYH)JUMP=1
      GO TO 75
C
C     ACKNOWLEDGE NO MORE CELLS
C
   70 NOTOK=.TRUE.
C
   75 IX=MIN0(NX,MAX0(1,IX))
      IY=MIN0(NY,MAX0(1,IY))
      RETURN
      END
```

12.0 HEIGHT OF A POINT (HEIGHT)

1. Purpose

This routine computes the altitude of a point given its X-Y coordinates and topographic data.

2. Arguments

INPUTS

XIN	-	X coordinate of point
YIN	-	Y coordinate of point

OUTPUTS

HEIGHT	-	A function that returns altitude at (X,Y)

3. Procedure

This routine uses a double linear interpolation scheme to obtain the height at any point in the tabular data base.

4. Comments

None.

5. Flow Chart

See Figure 8.

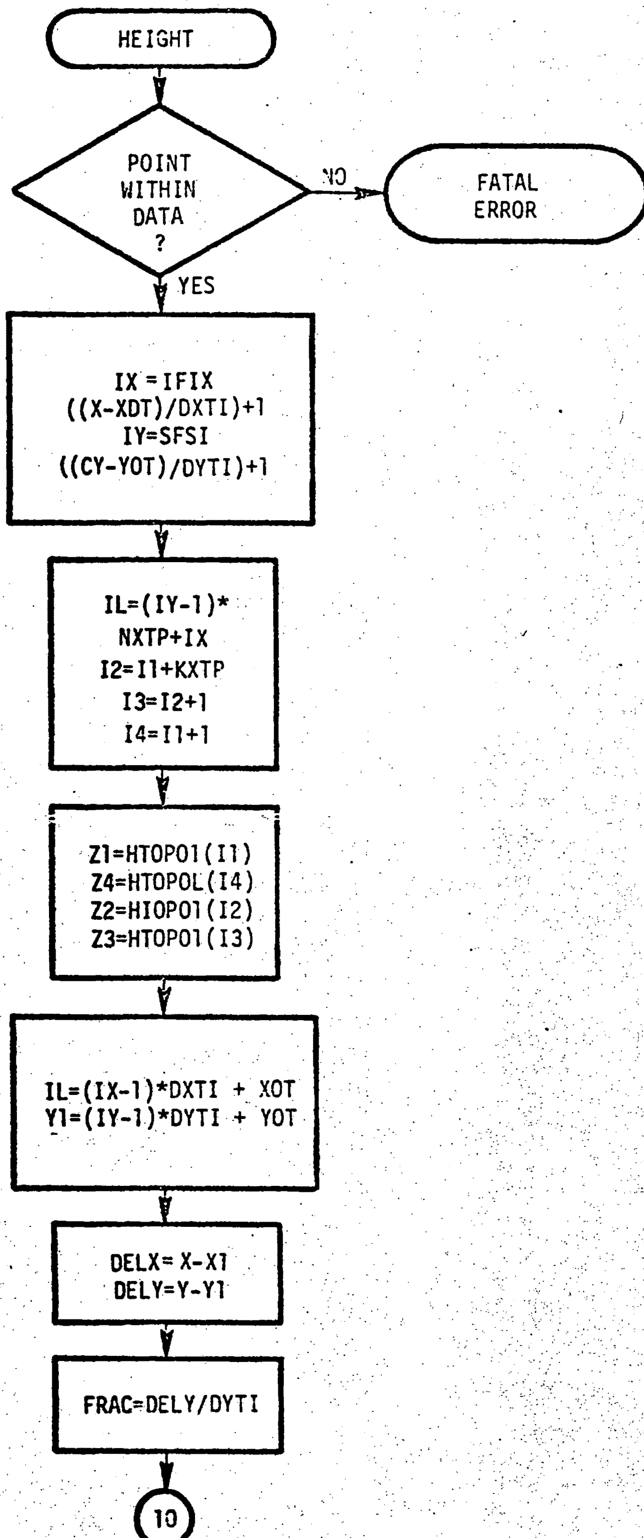


FIGURE 8 LOGIC STRUCTURE FOR ROUTINE HEIGHT

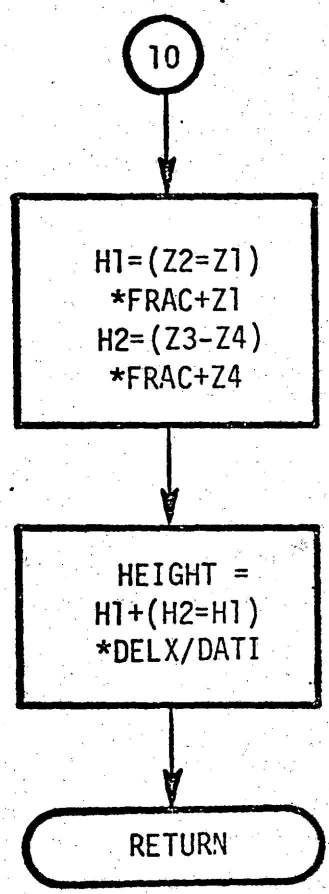


FIGURE 8 (CONTINUED)

```
      FUNCTION HEIGHT(XIN,YIN)
C
C     THIS ROUTINE COMPUTES THE ALTITUDE OF A POINT GIVEN
C     THE X-Y COORDINATES AND TOPOGRAPHIC DATA.
C     VER 3.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     XIN     - X COORDINATE OF POINT
C     YIN     - Y COORDINATE OF POINT
C         OUTPUTS
C     HEIGHT - FUNCTION RETURNING ALTITUDE AT (X,Y)
C
C         ROUTINES USED
C     FLOAT, IFIX
C
      COMMON/TOPO2/KUADT,XOT,YOT,XMT,YMT,DXTI,DXTII,DYTI,DYTII,SRUF,
     &             DXTO2,DYTO2,NXTP,NYTP,NPTST,HBC(51,51)
      COMMON /WARNS/ IWARN(3)
C
C     ************************
C
C     IS POINT WITHIN DATA
C
    5 IF(XOT.LE.XIN .AND. XIN.LE.XMT
     &              .AND.
     &   YOT.LE.YIN .AND. YIN.LE.YMT)GO TO 16
      XOUT=XIN*3.28084
      YOUT=YIN*3.28084
      IF(IWARN(1).EQ.0)PRINT     2001,XOUT,YOUT
      IWARN(1)=1
C
   16 HEIGHT=0.0
      IF(KUADT.EQ.0)GO TO 15
C     LOCATE LOWER LEFT CORNER
C
      IX=IFIX((XIN-XOT)*DXTII)+1
      IX=MIN0(IX,NXTP-1)
      IX=MAX0(IX,1)
      IY=IFIX((YIN-YOT)*DYTII)+1
      IY=MIN0(IY,NYTP-1)
      IY=MAX0(IY,1)
C
C     FIND HEIHGT OF SURROUNDING POINTS
C
      Z1=HBC(IX,IY)
      Z2=HBC(IX,IY+1)
      Z3=HBC(IX+1,IY+1)
      Z4=HBC(IX+1,IY)
C
C     PREFLEM DOUBLE LINEAR INTERPOLATION
C
      X1=FLOAT(IX-1)*DXTI+XOT
      Y1=FLOAT(IY-1)*DYTI+YOT
      DELX=AMAX1(0.0,AMIN1(DXTI,XIN-X1))
      DELY=AMAX1(0.0,AMIN1(DYTI,YIN-Y1))
      FRAC=DELY*DYTII
      H1=(Z2-Z1)*FRAC+Z1
      H2=(Z3-Z4)*FRAC+Z4
      HEIGHT=H1+(H2-H1)*DELX*DXTII
C
   15 RETURN
C
C     INCORRECT X,Y
C
 2001 FORMAT(3X,44H***WARNING*** DATA BASE EXCEEDED-HEIGHT-X,Y=,
     & 2E15.6)
      END
```

13.0 SWIND Version 1.00 26 November 1975

1. PURPOSE

This subroutine calculates wind velocity at fuel height in terms of geostrophic wind velocity, by a variation of an Ekman spiral.

2. ARGUMENTS

INPUT

WS850	=	wind speed at 850 MB (m/min)
WAZ850	=	wind azimuth at 850 MB (rad)
TMET	=	time of met data (min. after midnight)
TOD	=	time of day at which wind velocity is to be calculated (min. after midnight)
HSUR	=	height of surface above sea level (m)
DEPTH	=	fuel height (m)

OUTPUT

WVEL	=	wind speed at fuel height (m/min)
WAS	=	wind azimuth at fuel height (rad)

3. PROCEDURE

The following form of an Ekman spiral is given in Reference 3

$$V_p = |\vec{V}_q| \; (1-e^{-\gamma z}\cos\gamma z)$$
$$V_\rho = |\vec{V}_q| e^{-\gamma z}\sin\gamma z$$

where $V_{geo} = |\vec{V}_q|$, and V_p and V_ρ are the components of the wind speed at height z above the surface, parallel and perpendicular to $\vec{V}_q$ respectively,

$\vec{V}_q$ = wind velocity at 850 MB

γ = $(f/2k)^{1/2} \simeq 3.16 \times 10^{-3} m^{-1}$

$f \quad = \quad$ coriolis factor $\simeq 10^{-4}sec^{-1}$ at 45° lat.

$k \quad = \quad$ eddy viscosity $\simeq 5m^2/sec.$

The wind speed and direction at height z above the surface are then given by

$$V(z) = (V_p^{\ 2} + V_\rho^{\ 2})^{1/2}$$

$$\alpha(z) = \alpha_q - \tan^{-1}(V_\rho/V_p).$$

It can be seen that the above Ekman spiral estimates wind velocity at a height z above the surface in terms of the geostrophic wind velocity and the height above the surface, independent of the surface height and surface cover depth.

For the present purposes, the important quantity is wind velocity at fuel height, which is expected to depend upon both the height of the surface and the height of surface cover. Accordingly, the Ekman spiral has been modified to include these variables by considering two regimes; a surface regime and a cover regime. The wind in the surface regime is expressed in terms of the geostrophic wind as

$$W_{SE}(z) = W_{GE}(1.0-e^{-\gamma z}\cos\gamma z) - W_{GN}e^{-\gamma z}\sin\gamma z$$

$$W_{SN}(z) = W_{GN}(1.0-e^{-\gamma z}\cos\gamma z) + W_{GE}e^{-\gamma z}\sin\gamma z$$

$$W_S(z) = (W_{SE}^{\ 2} + W_{SN}^{\ 2})^{1/2}$$

$$\alpha_S(z) = \alpha_G - \tan^{-1}(W_{SE}/W_{SN})$$

where

$z \quad = \quad H_S + D =$ height of top of cover regime,

$H_X \quad = \quad$ local surface height above sea level (m)

$D \quad = \quad$ local cover depth

$t \quad\quad$ time of day (hours from midnight)

The quantity γz is defined by

$$\gamma z = 0.00316\ H_S(0.475 + 0.825(1 + \cos(t-17)))$$

The wind speed at mid-cover height is defined by

$$W_F = 2.0\ W_S\ (1.0-e^{-0.50})$$

where the mid-cover wind speed is doubled in anticipation of it being halved in the spread rate model, hence the notation of wind speed at fuel height in the figures.

4. MODEL RESULTS

Sample results from SWIND are presented in Figures 9 through 11, which are described below. Figure 9 shows the wind speed at fuel height plotted against surface height for five values of geostrophic wind speed, where geostrophic wind azimuth angle (A_g), calculation time (T) and cover depth (D) are held constant at the indicated values. It will be seen that under the conditions of Figure 9 wind speed at fuel height increases monotonically with surface height and with geostrophic wind speed.

Figure 10 shows wind speed at fuel height plotted against calculation time for four values of surface height, where geostrophic wind speed (W_g), geostrophic wind azimuth angle (A_g) and cover depth (D) are held constant. It will be seen that for the conditions of Figure 10 windspeed at fuel height exhibits a quasi-sinusoidal diurnal variation, with a monotonic increase with increasing surface height. Figure 10a shows wind speed at fuel height plotted against cover depth for four values of surface height, where geostrophic wind velocity (W_g,A_g) and calculation time are held constant. Within the limits of cover depth common to Southern California fuels Figure 10a shows wind speed at fuel height to increase almost linearly with increasing fuel height, and to increase quasi-logrithmically with increasing surface height.

Figure 9. Wind Speed at Fuel Height Versus Surface Height for Five Values

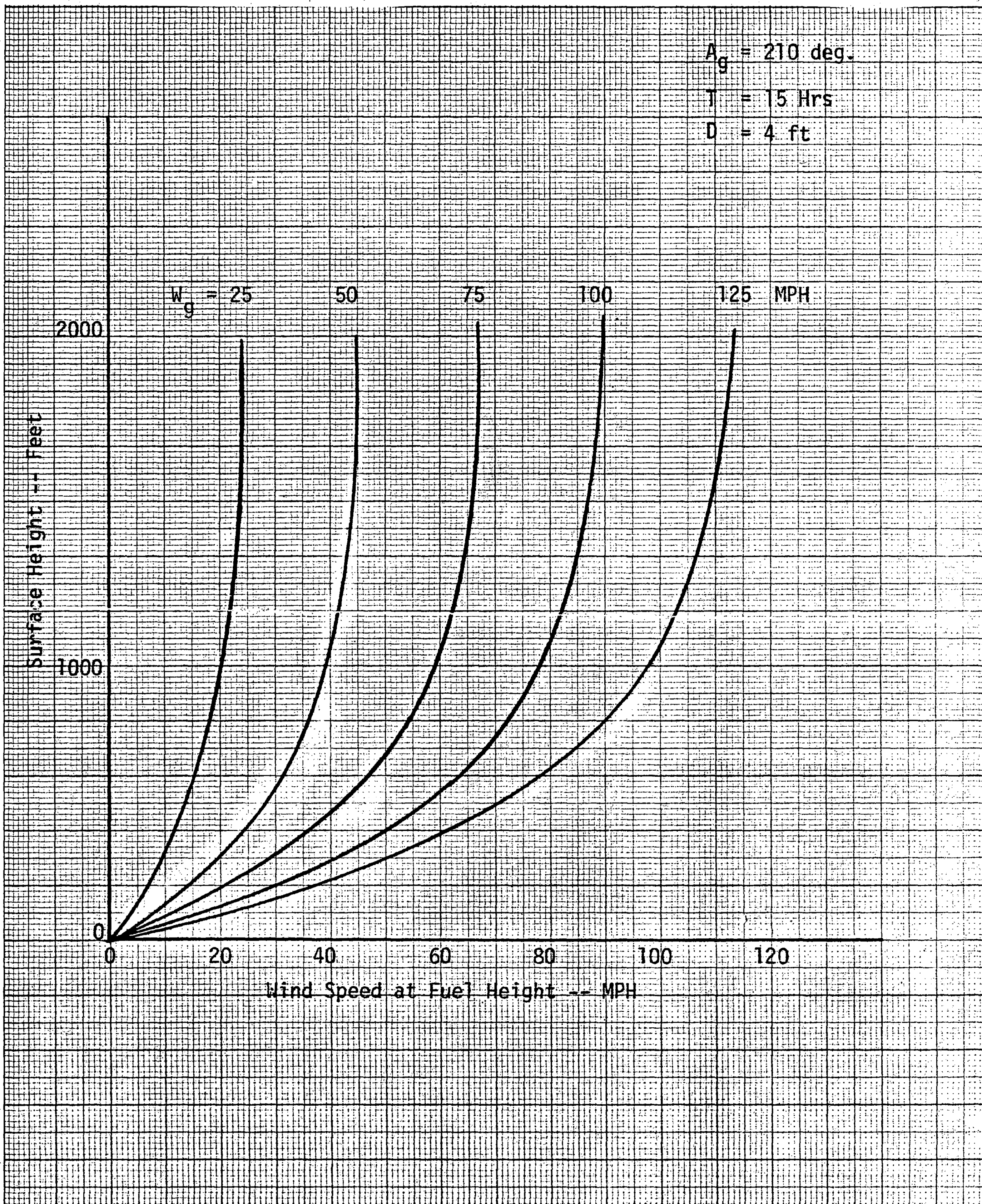


Figure 10. Wind Speed at Fuel Height Versus Time of Day for Four Values of Surface Height

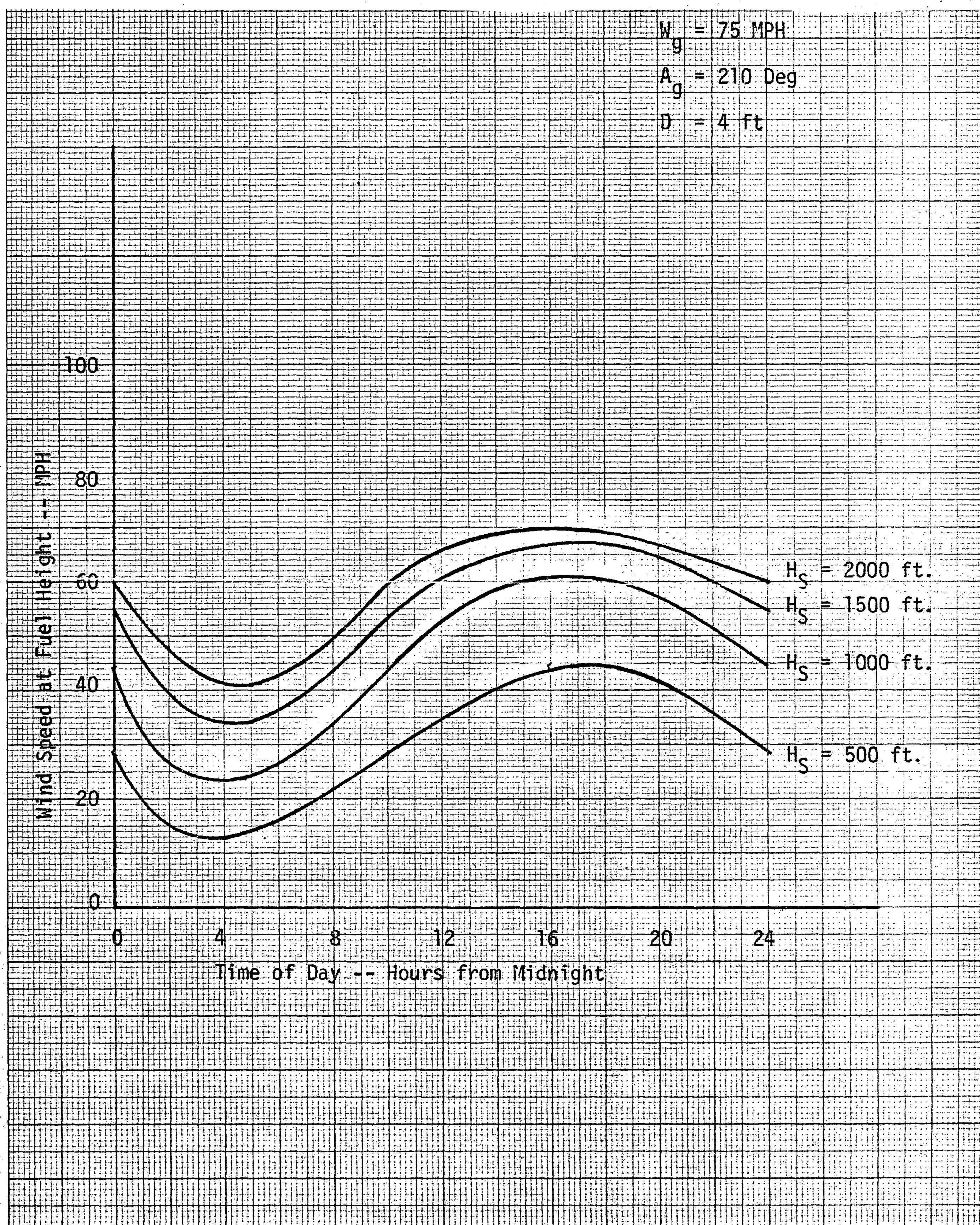


Figure 10A. Wind Speed at Fuel Height Versus Fuel Height for Four Values of Surface Height

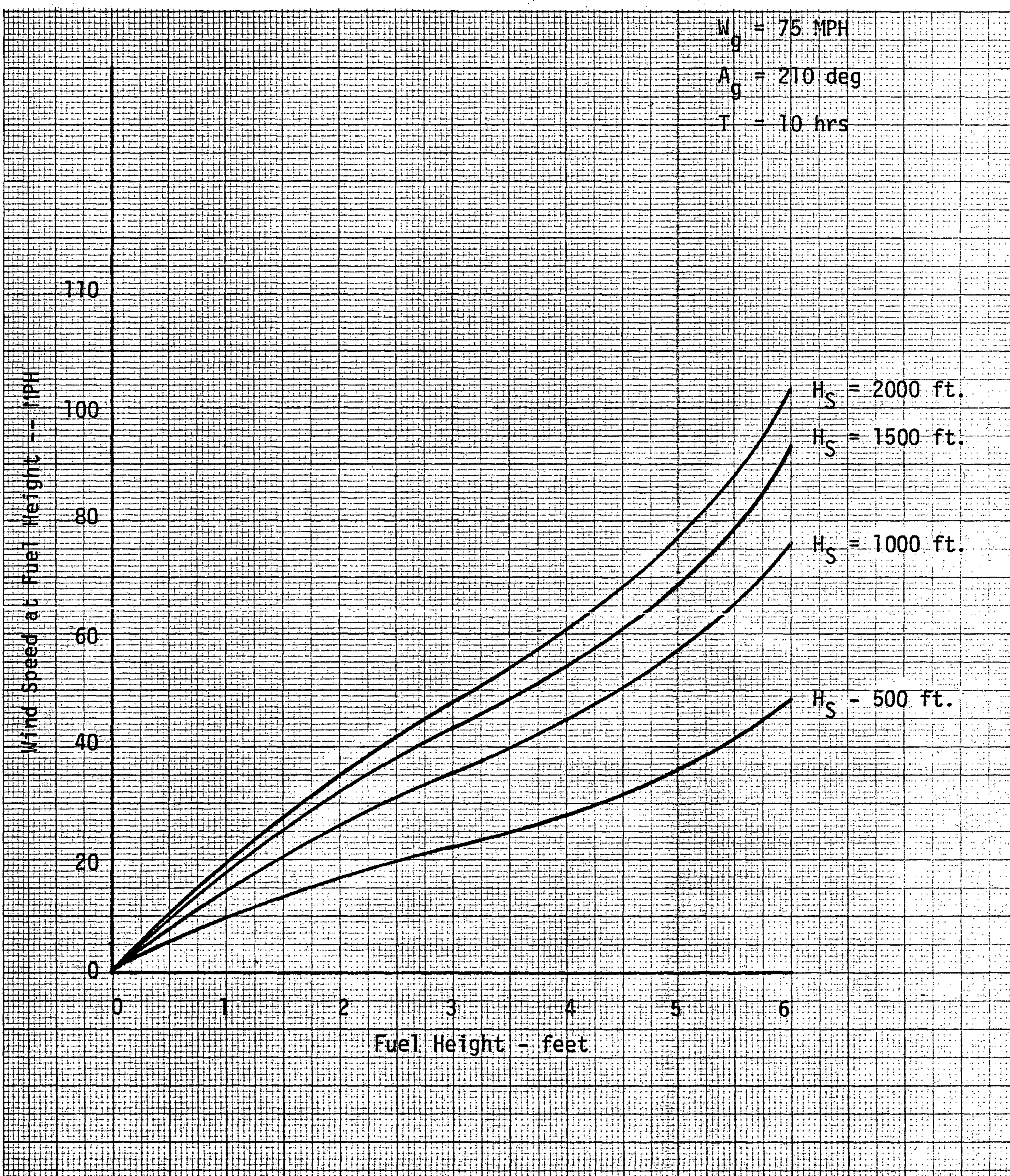
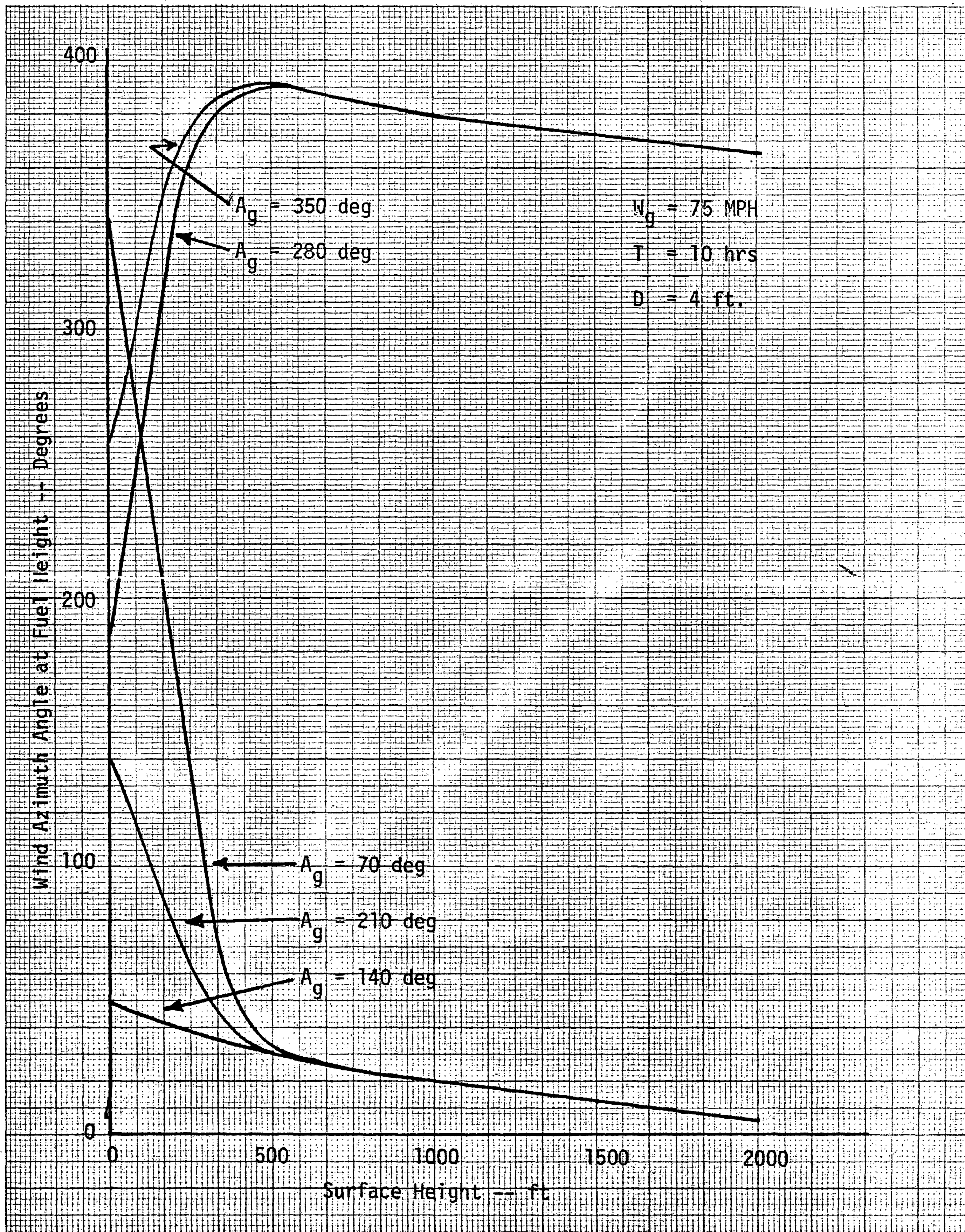


Figure 11 shows wind azimuth angle at fuel height plotted against surface height for five values of geostrophic wind azimuth angle. It can be seen that SWIND estimates the wind direction to change at a uniform rate of about 6 degrees/100 feet down to a surface height of about 500 ft. For surface heights between about 500 feet and sea level the estimated rate of change of wind direction is from about 4 to 60 degrees/100 feet, depending upon geostrophic wind azimuth angle.

5. Flow Chart

Not Required

Figure 11. Wind Azimuth Angle at Fuel Height Versus Surface Height for Five Values of Geostrophic Wind Azimuth Angle



```
      SUBROUTINE SWIND(WS850,WAZ850,TMET,TOD,HSUR,DEPTH,WVEL,WAZ)
C
C     THIS ROUTINE CALCULATES SURFACE WIND FROM GEOSTROPHIC WIND
C     USING AN EKMAN SPIRAL
C     VER 2.00 04/21/76   (VER 1.00 11/26/75)
C
C     PROGRAMMER - DR. JAMES SANDERLIN
C
C          INPUTS
C     WS850  - WIND SPEED AT 850 MB (METERS/MIN)
C     WAZ850 - WIND AZIMUTH AT 850 MB (RADIANS)
C     TMET   - TIME OF MET. DATA (MINUTES FROM MIDNIGHT)
C     DEPTH  - DEPTH AT WHICH WIND IS TO BE CALCULATED (METERS)
C              (ONE-HALF FUEL HEIGHT)
C     TOD    - TIME OF DAY OF MET. DATA DESIRED (MINUTES FROM MID.)
C     HSUR   - HEIGHT OF SURFACE ABOVE SEA LEVEL (METERS)
C          OUTPUTS
C     WVEL   - WIND VELOCITY AT DEPTH METERS ABOVE SURFACE (METERS/MIN)
C     WAS    - WIND AZIMUTH AT DEPTH ABOVE SURFACE (RADIANS)
C
C          EAST, NORTH COMPONENTS OF GEOSTROPHIC WIND
      TMETH=TMET/60.0
      TODH=TOD/3438.0
      WSGE = WS850*SIN(WAZ850)
      WSGN = WS850*COS(WAZ850)
C          GAMMAZ AT TOP OF SURFACE LAYER
      H0=0.5+15.25*(1.0+COS(0.2618*(TODH-14.0)))
      Z0=HSUR+H0
      GAMAZ=Z0*(7.7E-3+7.55E-3*SIN(0.2618*TMETH+3.4034))
C
C     EAST, NORTH COMPONENTS OF SURFACE WIND
C
      EGZ   = EXP(-GAMAZ)
      COSGZ= COS(GAMAZ)
      SINGZ= SIN(GAMAZ)
      WSSE = WSGE*(1.0-EGZ*COSGZ)-WSGN*EGZ*SINGZ
      WSSN = WSGN*(1.0-EGZ*COSGZ)+WSGE*EGZ*SINGZ
C          WIND AT TOP OF BOUNDARY LAYER
      WVEL= WS850*(1.0-EGZ)
      WAZ  = ATAN2(WSSE,WSSN)
C          WIND AT FUEL HEIGHT
      WVEL=WVEL*(1.0-EXP(-0.38*DEPTH))
      RETURN
      END
```

14.0 EDIT RESOURCES (EDIRES)

1. Purpose

This routine allows an interactive user to enter new resources or to alter existing resources.

2. Arguments

INPUTS

RFLTYP	-	An alpha variable describing the resource file type
		RFLTYP = OLD Resources are to be read from logical unit 3
		RFLTYP = NEW Resources are to be input and recorded on logical unit 3.
IYRF	-	Year of fire (years-1900)

NOTE: RESNTR is used as an input routine.

INPUT/OUTPUTS FROM COMMON/ROUT12/

NRSTØT	-	Total number of resource entries
IDR	-	Resource type index array
NØRS	-	Resource quantity array
ETA	-	Resource estimated time of arrival
ACT	-	Resource air cycle time
IORD	-	Resource time order array
NRSMAX	-	Maximum number of resources which may be stored.

3. Procedure

A listing of the resource table is printed and the interactive user is able to add, delete or change lines as required. When all corrections

are completed, a pointer array (IORD) is generated so that resource data may be accessed according to time of arrival. The data are then written on logical unit 3.

4. Comments

If any previous data existed on logical unit 3, they are overwritten.

5. Flow Chart

See Figure 12.

Figure 12 Edit Resources

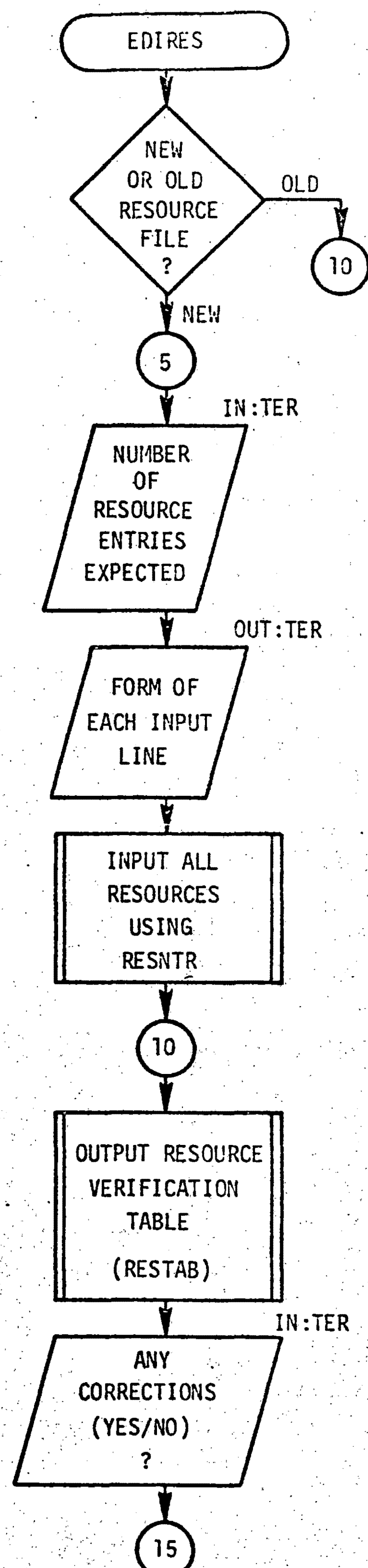


Figure 12 Edit Resources (Cont'd)

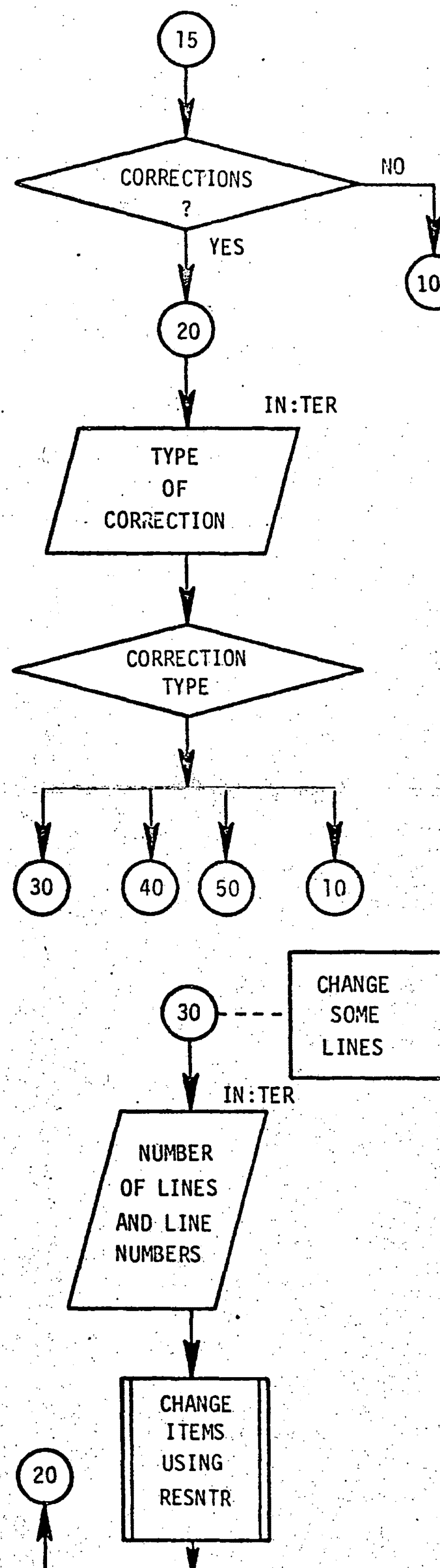
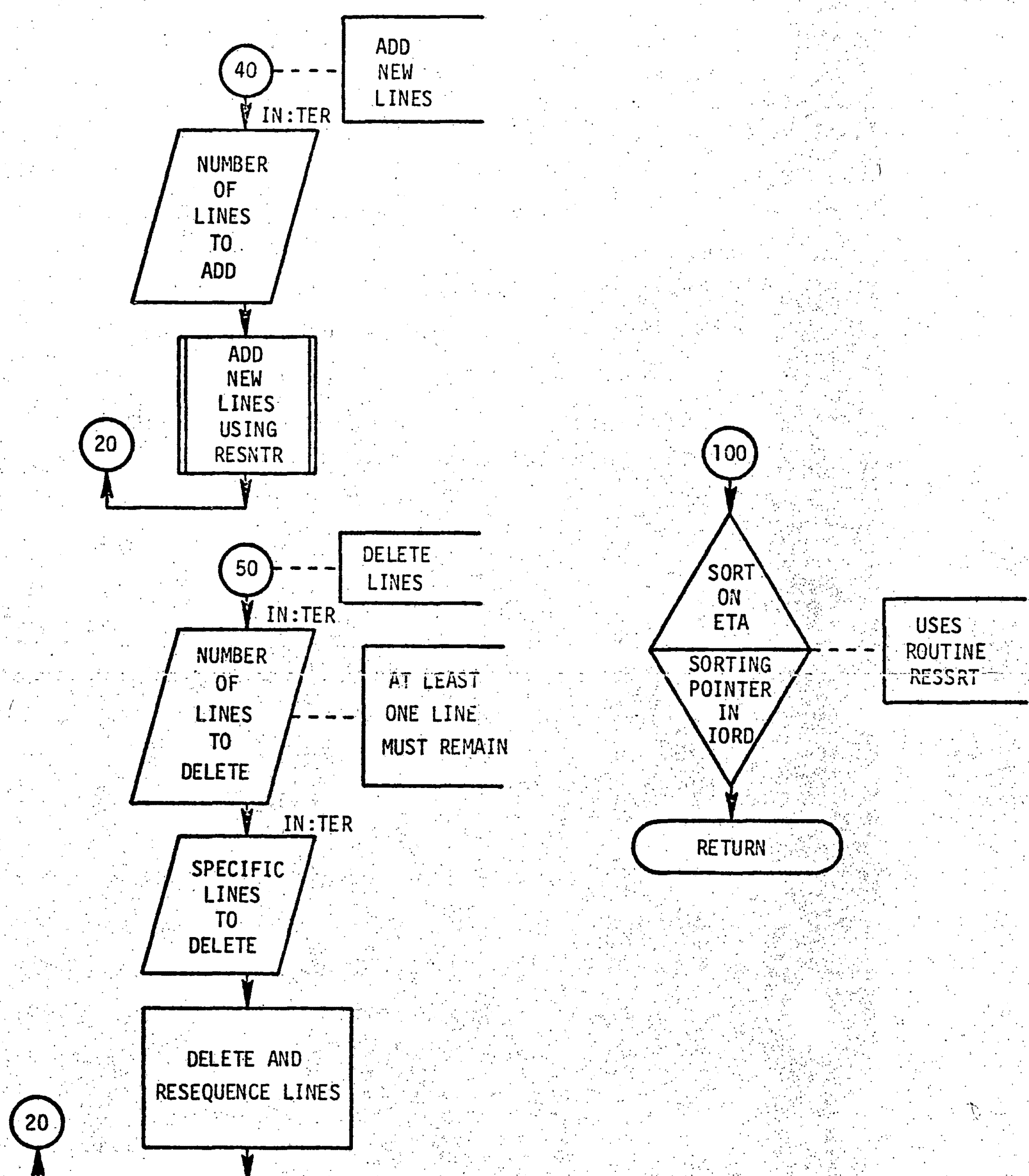


Figure 12 Edit Resources (Cont'd)



```
$$$$$EDIRES
10000        $ LINK UFILE1
10001        SUBROUTINE FILE1
10002C       PRINT," FILE 1 NOW LOADED"
10003        RETURN
10004        END
10010        SUBROUTINE EDIRES(RFLTYP,IYRF)
10020C
10030C       THIS ROUTINE INPUTS AND EDITS RESOURCES
10040C       VER 1.00 (FOR USE ON G. E. TIMESHARE ONLY)
10050C
10060C       PROGRAMMER - JOHN SUNDERSON, JR.
10070C
10080C          INPUTS FROM LIST
10090C       RFLTYP - AN ALPHA VARIABLE DESCRIBING THE RESOURCE TYPE
10100C                 OLD=RESOURCES ARE TO BE READ FROM LOGICAL UNIT 3
10110C                 NEW=RESOURCES ARE ALL TO BE ENTERED THROUGH THE
10120C                      USER TERMINAL
10130C       IYRF   - YEAR OF FIRE (YEARS-1900)
10140C          INPUT ROUTINES USED (QUANTITIES ARE LISTED IN EACH ROUTINE)
10150C       RESNTR(VER 2.00)
10160C          OUTPUTS TO MEMORY
10170C       AN UPDATED AND SORTED (ON ETA) AVAILABLE RESOUREC LIST (/ROUT12/)
10180C          OUTPUTS TO USER
10190C       A COPY OF THE RESOURCE TABLE IS PRINTED AS INPUT
10200C          INPUT/OUTPUTS FROM COMMON /ROUT12/
10210C       NRSTOT - TOTAL NUMBER OF RESOURCE ENTRIES
10220C       IDR    - RESOURCE TYPE INDEX ARRAY
10230C       NORS   - RESOURCE QUANTITY ARRAY
10240C       ETA    - RESOURCE ESTIMATED TIME OF ARRIVAL
10250C       ACT    - RESOURCE AIR CYCLE TIME
10260C       IORD   - RESOURCE TIME ORDER ARRAY
10270C          OUTPUTS TO COMMON /ROUT12/
10280C       NRSMAX - MAXIMUM NUMBER OF RESOURCES WHICH MAY BE STORED
10290C
10300C NOTE... IN ALL CASES THE RESOURCE DATA FILE (LOGICAL UNIT 3) WILL
10310C          BE UPDATED WITH A COPY OF THE RESOURCES INPUT
10320C
10330        ALPHA RFLTYP,NEW,OLD,ANS,YES,NO,CHAN,ADD,DELE,ENDC
10340C
10350        DIMENSION LINE(20)
10360C
10370        COMMON /ROUT12/ NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
10380       &                ETA(20),ACT(20)
10382        COMMON/BREKIT/NSTEP,IQSTEP,IMSTEP,IPSTEP(12),
10384       & ICFLAG(12),IMMSTP(12)
10390C
10400        DATA NEW/3HNEW/,OLD/3HOLD/,YES/3HYES/,NO/2HNO/,CHAN/4HCHAN/
10410        DATA ADD/3HADD/,DELE/4HDELE/,ENDC/3HEND/
10420C
10430C
10435        IF(ICFLAG(IQSTEP).NE.0)GO TO 10
10440        NRSMAX=20
10445        ICFLAG(IQSTEP)=1
10450C
10460C       IS A NEW RESOURCE TABLE TO BE INPUT OR AN OLD ONE EDITED
10470C
```

```
      IF(RFLTYP.EQ.OLD)GO TO 10
C           INPUT NEW TABLE
    5 PRINT    1000
 1000 FORMAT(3X,41HENTER EXPECTED NUMBER OF RESOURCE ENTRIES)
      PRINT    1003
 1003 FORMAT(7X,4H....)
      READ     2000,NRSTOT
 2000 FORMAT(V)
      IF(NRSTOT.LE.NRSMAX)GO TO 7
      PRINT    1001,NRSMAX
 1001 FORMAT(3X,40H***ERROR*** TOO MANY ENTRIES, MAXIMUM = ,I3)
      GO TO 5
    7 PRINT    1004
 1004 FORMAT(3X,49HENTER DATA FOR EACH RESOURCE AS A LINE CONTAINING/
     &         3X,28HRESOURCE NAME, QUANTITY, ETA/
     &         3X, 9HFORMAT IS/
     &        12X,22HNNNNNNNNNQQQHHMM-MMDDYY)
      NRSTO=NRSTOT
      NRSTOT=0
      CALL RESNTR(1,NRSTOT,IYRF)
      NRSTOT=NRSTO
C
C     EDIT RESOURCE ENTRIES
C
C           PRINT RESOURCE TABLE FOR REVIEW
   10 CALL RESTAB(IYRF)
      IF(NRSTOT.LE.0)GO TO 5
      RFLTYP=OLD
C
C     ARE ANY CORRECTIONS TO BE MADE
C
   12 PRINT    1005
 1005 FORMAT(/3X,34HARE THERE ANY CORRECTIONS TO BE MA,
     & 34HDE TO THE RESOURCE TABLE (YES/NO)   )
      PRINT    1003
      READ     2001,ANS
      IF(ANS.EQ.YES)GO TO 20
      IF(ANS.EQ.NO)GO TO 100
      PRINT    1006
 1006 FORMAT(3X,47H***ERROR*** INVALID ANSWER - USE ONLY YES OR NO)
      GO TO 12
C
C     FIND OUT WHAT KIND OF CORRECTIONS
C
   20 CALL BREAK(1)
      PRINT    1007
 1007 FORMAT(3X,63HENTER TYPE OF CORRECTION (CHANGE/ADD/DELETE/END OF CO
     &RRECTIONS))
      PRINT    1003
      READ     2001,ANS
 2001 FORMAT(A4)
      IF(ANS.EQ.CHAN)GO TO 30
      IF(ANS.EQ.ADD)GO TO 40
      IF(ANS.EQ.DELE)GO TO 50
      IF(ANS.EQ.ENDC)GO TO 10
      PRINT    1008
 1008 FORMAT(3X,56H****ERROR*** INVALID ANSWER - ENTER ONLY INDICATED CH
     &OICES)
      GO TO 20
C
C     CHANGE SOME LINES
C
```

```
   30 PRINT     1009
 1009 FORMAT(3X,32HENTER NO. OF LINES TO BE CHANGED)
      PRINT     1003
      READ     2000,NLINE
      IF(NLINE.EQ.0)GO TO 20
      NLINE=MIN0(NLINE,NRSTOT)
      PRINT     1014
      PRINT     1003
      READ     2000,(LINE(K),K=1,NLINE)
C          REINPUT ALL LINES WITH VALID LINE NUMBERS
      PRINT     1004
      NV=0
      DO 34 I=1,NLINE
      LIN=LINE(I)
      IF(LIN.GT.NRSTOT)GO TO 33
      NV=NV+1
      CALL RESNTR(LIN,LIN,IYRF)
   33 CONTINUE
   34 CONTINUE
      IF(NV.EQ.0)PRINT     1010,NRSTOT
 1010 FORMAT(3X,65H***ERROR*** NO VALID LINE NUMBER ENTERED- ALL MUST BE
     & LESS THAN ,I3)
      GO TO 20
C
C     ADD NEW RESOURCE ENTRIES
C
   40 PRINT     1011
 1011 FORMAT(3X,33HENTER NUMBER OF LINES TO BE ADDED)
      PRINT     1003
      READ     2000,NLAD
      NLAD=MIN0(NLAD,NRSMAX-NRSTOT)
      PRINT     1004
      CALL RESNTR(NRSTOT+1,NRSTOT+NLAD,IYRF)
      NRSTOT=NRSTOT+NLAD
      IF(NLAD.EQ.0)PRINT     1012,NRSMAX
 1012 FORMAT(3X,49H***ERROR*** RESOURCE TABLE FULL - MAX. ENTRIES = ,I3)
      GO TO 20
C
C     DELETE ENTRIES
C
   50 PRINT     1013
 1013 FORMAT(3X,32HENTER NO. OF LINES TO BE DELETED)
      PRINT     1003
      READ     2000,NLINE
      NLINE=MIN0(NLINE,NRSTOT-1)
      IF(NLINE.LE.0)GO TO 20
      PRINT     1014
 1014 FORMAT(3X,36HENTER LINE NUMBERS SEPERATED BY COMMAS)
      PRINT     1003
      READ     2000,(LINE(K),K=1,NLINE)
C          REMOVE INVALID LINE NUMBERS
      CALL BREAK(0)
      NL=NLINE
   52 LIN=LINE(NL)
      IF(LIN.GT.0.AND.LIN.LE.NRSTOT)GO TO 54
      LINE(NL)=LINE(NLINE)
      NLINE=NLINE-1
   54 NL=NL-1
      IF(NL.GT.1)GO TO 52
      IF(NLINE.LE.0)GO TO 20
C          FLAG LINES FOR DELETION
      DO 55 I=1,NRSTOT
      IORD(I)=1
   55 CONTINUE
```

```
      DO 56 I=1,NLINE
      LIN=LINE(I)
      IORD(LIN)=-1
   56 CONTINUE
C          DELETE LINES
      NWP=0
      DO 58 I=1,NRSTOT
      IF(IORD(I).LT.0)GO TO 57
      NWP=NWP+1
      IDR(NWP)=IDR(I)
      NORS(NWP)=NORS(I)
      ETA(NWP)=ETA(I)
      ACT(NWP)=ACT(I)
   57 CONTINUE
   58 CONTINUE
      NRSTOT=NRSTOT-NLINE
      CALL BREAK(1)
      GO TO 10
C
C     SORT ON ETA
C
  100 CALL BREAK(0)
      CALL RESSRT
      ICFLAG(IQSTEP)=0
      IPSTEP(IQSTEP)=1
      IQSTEP=IQSTEP+1
      CALL BREAK(1)
      RETURN
      END
```

15.0 RESOURCE ENTRY (RESNTR)

1. Purpose

This routine enters a specified portion of the resource list, and performs nominal validity tests.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

This routine requests each resource input line from the user. Each line is checked for a valid resource type, a valid ETA (time and date) and to determine if an air cycle time is required. For cases where an air cycle time is required, this is also requested.

4. Comments

None.

5. Flow Chart

See Figure 13.

Figure 13 Resource Entry

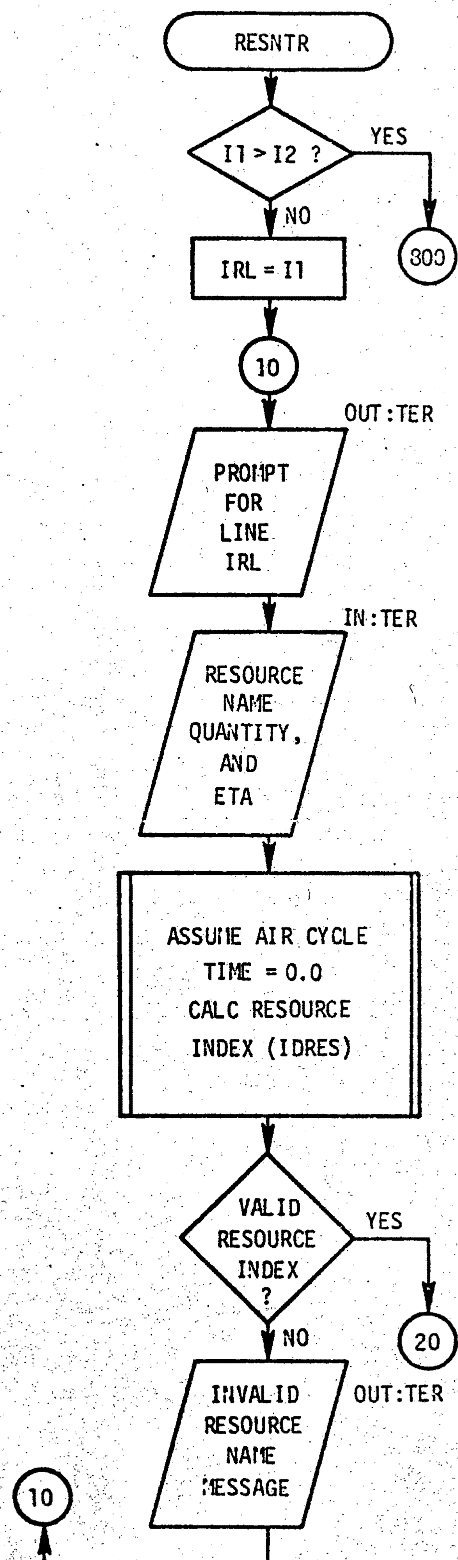


Figure 13 Resource Entry (Cont'd)

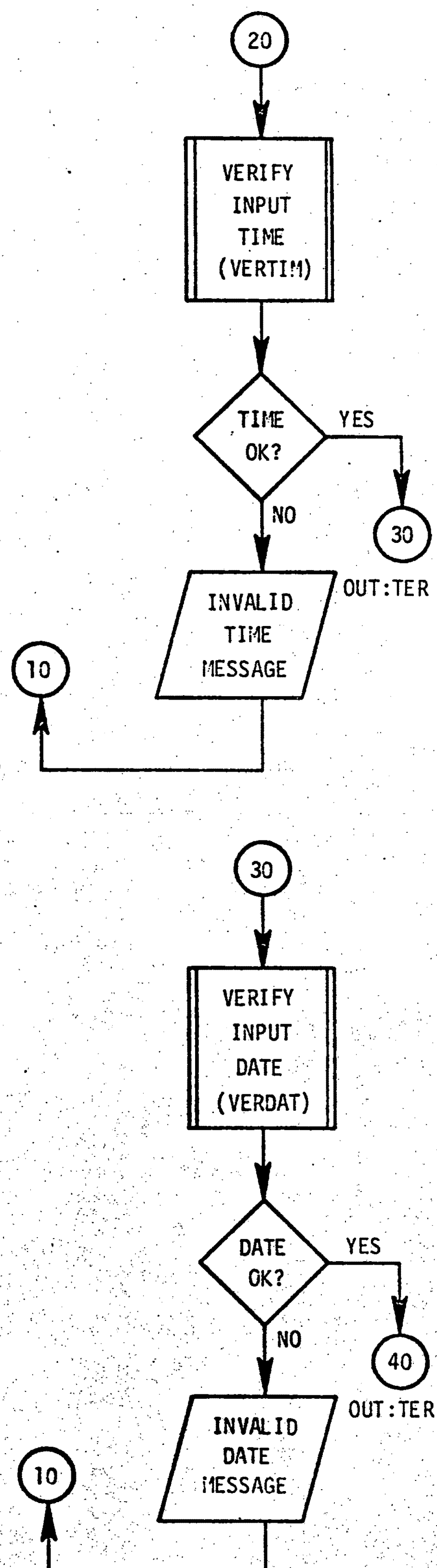
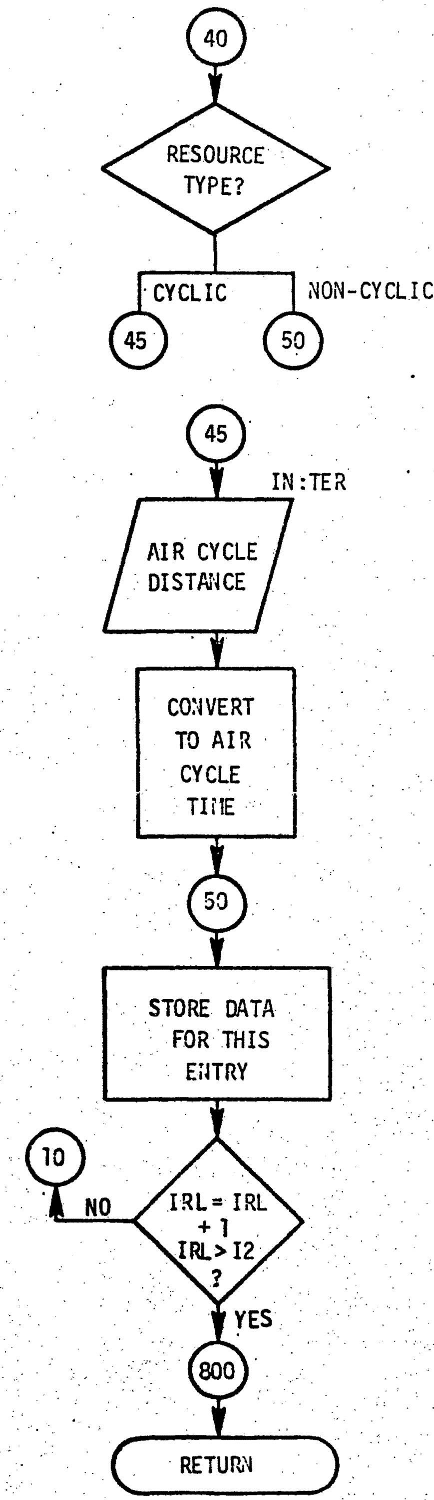


Figure 13 Resource Entry (Cont'd)



```
      SUBROUTINE RESNIP(I1,I2,IYRF)
C
C     THIS ROUTINE INPUTS AND PERFORMS SOME NOMINAL VALIDITY CHECKS
C     ON A SPECIFIED PORTION OF THE RESOURCE LIST
C     VER 2.00 (FOR USE ONLY ON G. E. TIMESHARE)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C          INPUTS FROM LIST
C     I1       - FIRST ENTRY ON RESOURCE TABLE TO BE INPUT
C     I2       - LAST  ENTRY ON RESOURCE TABLE TO BE INPUT
C     IYRF     - YEAR OF FIRE (YEARS-1900)
C          OUTPUTS TO COMMON /ROUT12/
C     IDR      - RESOURCE INDEX ARRAY
C     NORS     - NUMBER OF RESOURCES ARRAY
C     ETA      - EXPECTED (OR ACTUAL) TIME OF ARRIVAL ARRAY
C                (TIME IS IN MINUTES RELATIVE TO TOF)
C     ACT      - AIR CYCLE TIME (WHERE APPLICABLE, IN MINUTES)
C          INPUTS FROM INTERACTIVE USER
C     NAME     - RESOURCE ALPHANUMERIC IDENTIFIER (SEE RESOURCE
C                CHARACTERISTIC TABLE MRC-7512-6-1075 P.120)
C     QUANT    - QUANTITY OF EACH RESOURCE
C     ETAR     - EXPECTED (OR ACTUAL) ARRIVAL TIME OF
C                RESOURCE (FORM HHMM MMDDYY)
C     CY.DIS.- CYCLE DISTANCE FOR AIR RESOURCES IN MILES
C          INPUTS FROM COMMON /TIME1/
C     TOF      - TIME OF FIRE (MINUTES FROM MIDNIGHT)
C     JDOF     - DAY OF FIRE (DAYS FROM DEC 31)
C
      LOGICAL OK
      ALPHA NAM1,NAM2
C
      COMMON /ROUT12/ NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
     &            ETA(20),ACT(20)
      COMMON/TIME1/JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
      COMMON /RSDATA/ DECOD1(2,8),MAXR,MAXC,MAXS,DUM2(30),ID(15),
     &                         DUM1(300)
C
C     LOOP OVER THE REQUIRED RESOURCE ENTRIES
C
      IF(I1.GT.I2)GO TO 800
      DO 700 IRL=I1,I2
   10 PRINT    1000,IRL
 1000 FORMAT(4X,I3,4H....)
C
C     READ RESOURCE ENTRY SUPPLIED BY USER (FIXED FORMAT)
C
```

```
      READ     2000,NAM1,NAM2,IQUANT,IETAT,IETADT
 2000 FORMAT(A4,A4,I3,I4,1X,I6)
      IDR1=IDRESU(NAM1,NAM2)
      IF(IDR1.NE.0)GO TO 20
      PRINT    1001
 1001 FORMAT(3X,48H***ERROR*** INVALID RESOURCE NAME - RETYPE ENTRY)
      GO TO 10
C
C     CHECK ETA TIME AND DATE
C
   20 CALL VERTIM(IETAT,IETAH,IETAM,IETAMM,OK)
      IF(OK)GO TO 30
      PRINT    1002
 1002 FORMAT(3X,44H***ERROR*** INVALID ETA TIME - RETYPE ENTRY )
      GO TO 10
   30 CALL VERDAT(IETADT,IETAD,IETAM,IETAY,JETA,OK)
      IF(OK)GO TO 40
      PRINT    1004
 1003 FORMAT(7X,4H....)
 1004 FORMAT(3X,43H***ERROR*** INVALID ETA DATE - RETYPE ENTRY)
      GO TO 10
C
C     INPUT AIR CYCLE DISTANCE, IF REQUIRED, AND COMPUTE ACT
C
C                   ENG1 ENG2 ENG3 ENG4 HCT1 HCT2 DOZ1 DOZ2 RWHC1 FWRD1
   40 GO TO ( 50 , 50 , 50 , 50 , 50 , 50 , 50 , 50 , 50  , 45  ,
     &          45  , 50  , 50  ,   50,  50 ),IDR1
C                   RWWD1 FULR1 FULR2  TK1  TK2
   45 PRINT    1005,NAM1,NAM2
 1005 FORMAT(3X,42HENTER AIR CYCLE DISTANCE (MILES) FOR THIS ,2A4)
      PRINT    1003
      READ     2001,AIRCYD
 2001 FORMAT(V)
C          CONVERT TO METERS
      AIRCYD=AIRCYD*1609.3
C
C          CALCULATE AIR CYCLE TIME
C
      IF(IDR1.EQ.11)GO TO 47
C          CYCLE TIME FOR B-17
      ACT1=20.0+0.0005213*AIRCYD
      GO TO 50
C          CYCLE TIME FOR S-61
   47 ACT1=3.5+0.000224*AIRCYD
C
C     STORE DATA FOR THIS ENTRY
C
   50 IDR(IRL)=IDR1
      NORS(IRL)=IQUANT
      ETA(IRL)=FLOAT(IETAMM+(JETA-JDOF)*1440)-TOF
      ACT(IRL)=ACT1
C
C     LOOP TO NEXT ENTRY
C
  700 CONTINUE
  800 RETURN
      END
```

16.0 PRINT RESOURCE TABLE (RESTAB)

1. Purpose

This routine prints a tabulation of available resources.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

The data are converted to a suitable form for output and printed.

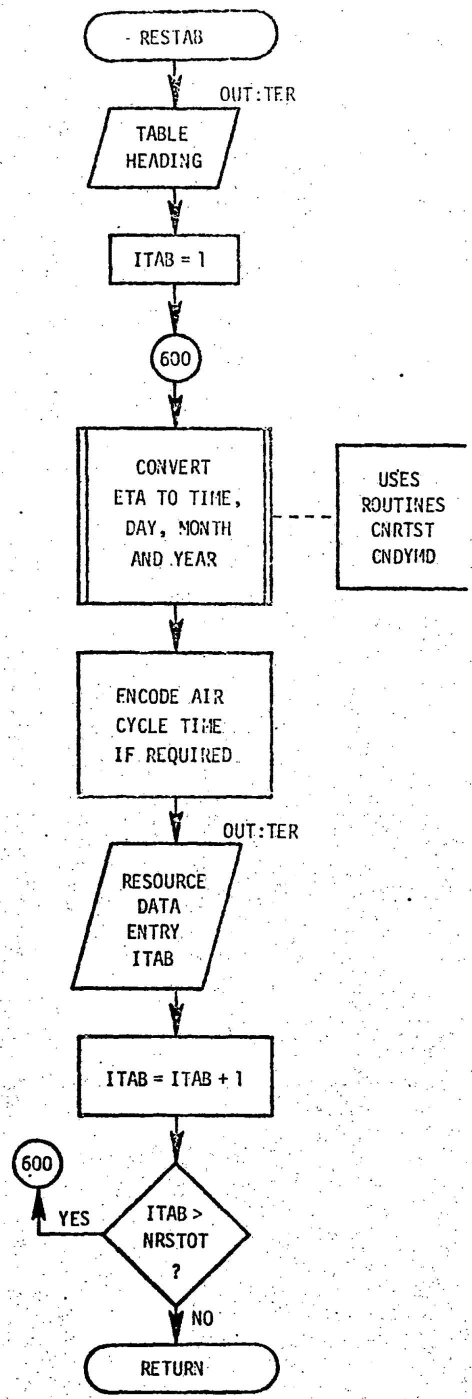
4. Comments

None.

5. Flow Chart

See Figure 14.

Figure 14 Print Resource Table



```
      SUBROUTINE RESTAB(IYRF)
C
C     THIS ROUTINE PRINTS THE AVAILABLE RESOURCE TABLE AS IT WAS INPUT
C     VER 1.00 (FOR USE ON G. E. TIMESHARE ONLY)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C          INPUT FROM LIST
C     IYRF    - YEAR OF FIRE (YEARS-1900)
C          INPUTS FROM COMMON /ROUT12/
C     NRSTOT  - TOTAL NUMBER OF AVAILABLE RESOURCE ENTRIES
C     IDR     - RESOURCE TYPE INDEX ARRAY
C     NORS    - RESOURCE QUANTITY ARRAY
C     ETA     - RESOURCE ESTIMATED TIME OF ARRIVAL
C     ACT     - RESOURCE AIR CYCLE TIME
C          INPUTS FROM COMMON /RSDATA/
C     NAME    - HOLLERITH RESOURCE IDENTIFIER
C          INPUTS FROM COMMON /TIME1/
C     TOF     - TIME OF FIRE (MINUTES FROM MIDNIGHT)
C     JDOF    - DAY OF FIRE (DAYS FROM DEC 31)
C          OUTPUTS
C     THIS ROUTINE OUTPUTS A PRINTED TABLE ON THE USER TERMINAL
C
      ALPHA NA,BLANK,COUT(2)
C
      COMMON/ROUT12/NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
     &              ETA(20),ACT(20)
      COMMON /RSDATA/ DECODI(2,2),MAXR,MAXS,MAXC,NAME(15,2),DUM(322)
      COMMON /TIME1/ JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
C
      DATA NA/3HN/A/,BLANK/1H /
C
 1000 FORMAT(3X,46HENTRY RESOURCE QUANTITY       ETA           CYCLE/
     &       3X,46H NO.     TYPE              TIME   DATE       TIME)
 1001 FORMAT(3X,I3,3X,2A4,3X,I3,3X,2I1,1H:,2I1,1X,I2,
     & 2(1H/2I1),1X,2A4)
C
C     OUTPUT TABLE HEADING
C
      PRINT    1003
 1003 FORMAT(19X,14HRESOURCE TABLE)
      IF(NRSTOT.LE.0)GO TO 700
      PRINT    1000
C
C     LOOP OVER TABLE ENTRIES
C
      DO 600 ITAB=1,NRSTOT
      ITYP=IDR(ITAB)
      IQUAN=NORS(ITAB)
      ETAR=ETA(ITAB)
      CALL CNRTST(TOF,JDOF,IYRF,ETAR,TMIN,NDAY,NYEAR)
      CALL CNDYMD(NDAY,NYEAR,NMON,NDOFM)
      IHRS=IFIX(TMIN/60.0)
      IMIN=IFIX(TMIN+0.00001)-IHRS*60
      COUT(1)=BLANK
```

```
      COUT(2)=NA
      IF(ITYP.EQ.10.OR.ITYP.EQ.11)ENCODE(COUT,1002)ACT(ITAB)
 1002 FORMAT(F8.0)
      IQHD1=IHRS/10
      IQHD2=IHRS-IQHD1*10
      IQMD1=IMIN/10
      IQMD2=IMIN-IQMD1*10
      IQDD1=NDOFM/10
      IQDD2=NDOFM-IQDD1*10
      IQYD1=NYEAR/10
      IQYD2=NYEAR-IQYD1*10
      PRINT    1001,ITAB,NAME(ITYP,1),NAME(ITYP,2),IQUAN,
     & IQHD1,IQHD2,IQMD1,IQMD2,NMON,IQDD1,IQDD2,IQYD1,
     & IQYD2,COUT
  600 CONTINUE
C
  650 RETURN
  700 PRINT    1005
 1005 FORMAT(3X,20HRESOURCE TABLE EMPTY)
      GO TO 650
      END
```

17.0 SORT RESOURCES (RESSRT)

1. Purpose

This routine sorts the resource list according to increasing ETA.

2. Arguments

<u>INPUTS</u>

NRSTØT	-	Number of resources
ETA	-	Expected time of arrival array

<u>OUTPUTS</u>

IORD	-	Order index array. Sequential elements give indices of resource entries in increasing ETA order.

3. Procedure

A basic ripple sort is used to reorder the IØRD array. The data in ETA are not reordered and these data are addressed indirectly from the IØRD array.

4. Comments

None.

5. Flow Chart

See Figure 15.

Figure 15 Sort Resources

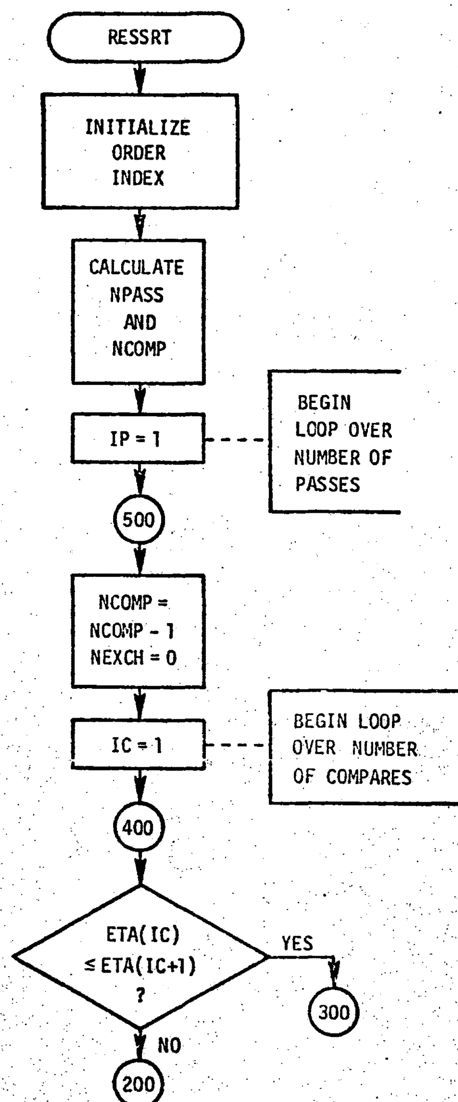
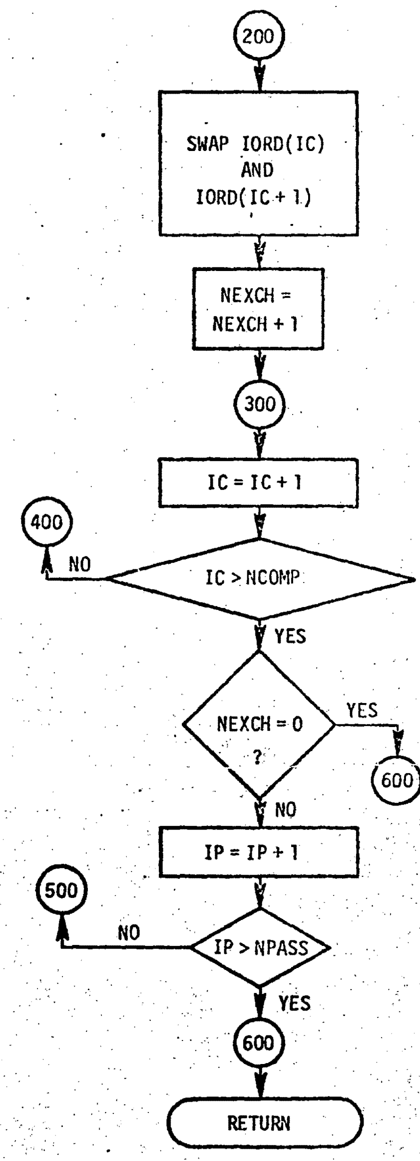


Figure 15 Sort Resources (Cont'd)



```
      SUBROUTINE RESSRT
C
C     THIS ROUTINE SORTS THE AVAILABLE RESOURCE TABLE BY ETA
C     VER 1.00 (FOR USE ON G. E. TIMESHARE ONLY)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS FROM COMMON /ROUT12/
C     NRSTOT - TOTAL NUMBER OF AVAILABLE RESOURCE ENTRIES
C     ETA    - EXPECTED TIME OF ARRIVAL IN MINUTES
C         OUTPUTS TO COMMON /ROUT12/
C     IORD   - ORDER INDEX.  SEQUENTIAL ELEMENTS GIVE INDICES OF
C                   RESOURCE ENTRIES WITH INCREASING ETAS.
C
      COMMON/ROUT12/NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
     &              ETA(20),ACT(20)
C
C     INITIALIZE IORD
C
      DO 10 I=1,NRSTOT
      IORD(I)=I
   10 CONTINUE
C
C     RIPPLE SORT ETAS USING INDIRECT ADDRESSING VIA IORD AND
C     ONLY REORDING IORD
C
      IF(NRSTOT.EQ.1)GO TO 600
      NPASS=NRSTOT-1
      NCOMP=NRSTOT
C
      DO 500 IP=1,NPASS
      NCOMP=NCOMP-1
      NEXCH=0
C
      DO 400 IC=1,NCOMP
      L1=IORD(IC)
      L2=IORD(IC+1)
C
      IF(ETA(L1).LE.ETA(L2))GO TO 300
C         SWAP ELEMENTS OF IORD
      IORD(IC)=L2
      IORD(IC+1)=L1
      NEXCH=NEXCH+1
C
  300 CONTINUE
  400 CONTINUE
      IF(NEXCH.EQ.0)GO TO 600
  500 CONTINUE
  600 CONTINUE
C
      RETURN
      END
```

18.0 LOCATE RESOURCE INDEX (IDRESØ)

1. Purpose

This function searches the resource characteristic table to locate the index of a specified resource type (alpha).

2. Arguments

INPUTS

NAM1	-	Alphanumeric resource name first half
NAM2	-	Alphanumeric resource name second half

OUTPUTS

IDRESØ	-	Resource index. If IDRESO = 0, the name was not located.

3. Procedure

This routine utilizes a sequential search.

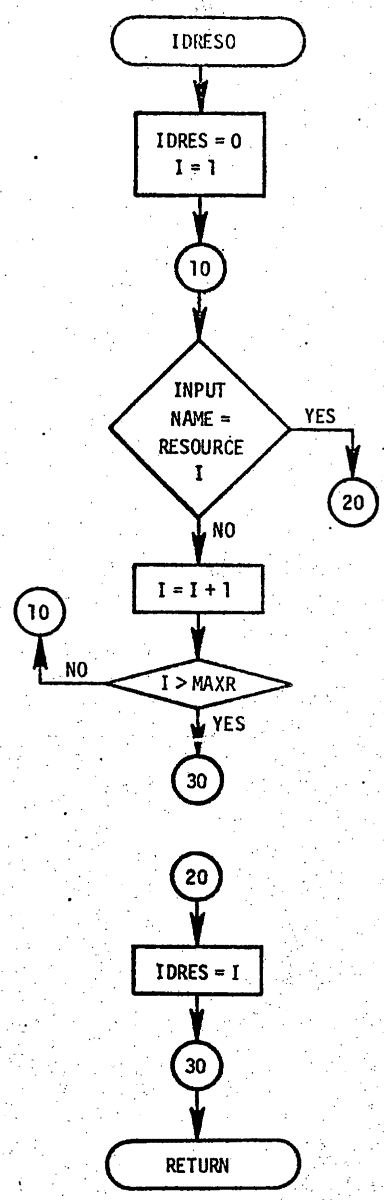
4. Comments

None.

5. Flow Chart

See Figure 16.

Figure 16 Locate Resource Index



```
      INTEGER FUNCTION IDRESO(NAM1,NAM2)
C
C     THIS ROUTINE SEARCHES THE RESOURCE CHARACTERISTIC TABLE FOR
C     THE INDEX OF THE RESOURCE GIVEN BY NAM1,NAM2
C     VER 1.00 (FOR USE ON G. E. TIMESHARE ONLY)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C          INPUTS FROM LIST
C     NAM1   - ALPHANUMERIC RESOURCE NAME 1ST HALF
C     NAM2   - ALPHANUMERIC RESOURCE NAME 2ND HALF
C          OUTPUT
C     IDRESO - FUNCTION CONTAINNING RESOURCE INDEX
C              =0 MEANS NAMED RESOURCE NOT FOUND
C
      ALPHA NAM1, NAM2,NAME
C
      COMMON /RSDATA/ DECODI(2,8),MAXR,MAXS,MAXC,NAME(15,2),DUM(1447)
C
      IDRES=0
      DO 10 II=1,MAXR
      I=II
      IF(NAM1.EQ.NAME(I,1).AND.NAM2.EQ.NAME(I,2))GO TO 20
   10 CONTINUE
      GO TO 30
   20 IDRES=I
   30 IDRESO=IDRES
      RETURN
      END
```

19.0 CALCULATE INITIAL PERIMETERS COMMAND PERI1

1. Purpose

This subroutine computes x,y perimeter points for initial perimeters at pre-determined times. The current version uses an elliptic spread model.

2. Arguments

Input - from COMMON

TOF	-	time of day (minutes from midnight) of fire report
JDOF	-	day of year of TOF (days from 31 December)
XF,YF	-	fire ignition point coordinates (meters)
AREA1	-	initial estimate of fire size (sq. meters)
PTIN	-	an array of default perimeter calculation times (in minutes from TOF)
NPTRQ	-	default number of entries in PTIN. Maximum allowed is 10.
ISPRED	-	spread model flag. Hardwired for elliptic model, ISPRED = 1.
NXY	-	the number of x,y points per perimeter. Maximum allowed is 20.

Output

ANGF	-	azimuth angle of orientation of fire in the elliptic model.
PT	-	an array containing the times at which perimeters are calculated (in minutes from TOF)
PX,PY	-	arrays containing the x,y points of each perimeter calculated.
NPT	-	the number of perimeters calculated. Maximum allowed is 10.

3. Procedure

The calculate initial perimeter routine is hardwired to the elliptic spread model. The perimeter calculation times are obtained from a default list of times (PTIN) which will have been previously set, either by the machine or interactively by the user.

The first perimeter time is zero; i.e., the time of fire reported. The first perimeter shape is taken to be a circle with the fire ignition point (x_f, y_f) at the center. A set of x,y points is computed for this circular perimeter. The radius (r) of the first perimeter is given by

$$r = \sqrt{\text{Area}/\pi}. \tag{1}$$

The azimuthal direction of the spreading fire (α_a) is then established by a call to the subroutine EDIRCT. This direction remains fixed for all the remaining perimeter calculation times in PERI1. The fire ignition point X_f, Y_f is at one of the foci of the elliptic perimeters. The orientation angle of the fire allows the propagation radii, r_a and r_b to be specified and the points x_a, Y_a and x_b, Y_b to be calculated as shown in Figure 17 by:

$$X_a = X_f + r_a \sin\alpha_a,$$

$$Y_a = Y_f + r_a \cos\alpha_a,$$

$$X_b = X_f + r_b \sin\alpha_b,$$

$$Y_b = X_f + f_b \cos\alpha_b.$$

The next perimeter time is selected and the time increment is computed. The perimeter is advanced in time independently from point A to point A' and point B to point B', by the subroutine ADVPPT. The new X_a', Y_a' and X_b', Y_b' points are used to calculate a new set of r_a' and r_b', which together with α_a are all that is needed to specify any point on the elliptic perimeter. The radii r_a', r_b' are calculated from

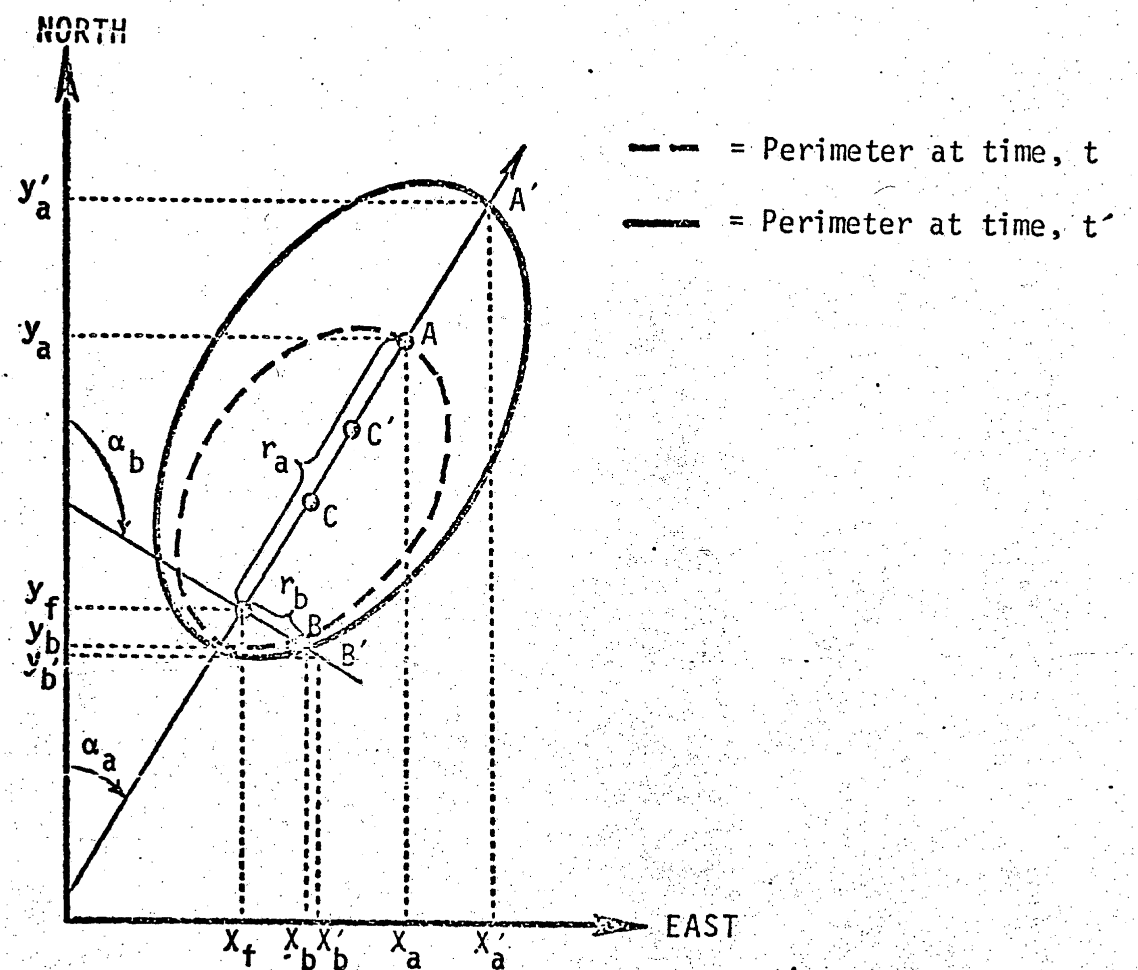


Figure 17. Elliptic spread model perimeters. Point F is the fire ignition point and points C and C´ locate the center of the ellipse at times t and t´. The other symbols are explained in the text.

$$r'_a = [(X'_a - X_f)^2 + (Y'_a - Y_f)^2]^{1/2},$$

$$r'_b = [(X'_b - X_f)^2 + (Y'_b - Y_f)^2]^{1/2}.$$

A set of perimeter points is then generated using the subroutine PPOINT and they are stored in the initial attack perimeter list (COMMON/PERIM1/). The above procedure is repeated until the initial perimeter calculation time list is exhausted.

The last perimeter computed by PERI1 is stored as the first perimeter of the campaign fire perimeter list (COMMON/PERIME/) together with the necessary elliptic model parameters α_a, α_b and X_b,Y_b. The X_a,Y_a points for all perimeters are stored as the first X,Y point for each perimeter.

4. Comments

The current version of the elliptic spread model does not allow the angle of the line of apsides to change. For evaluate initial attack, this is not too serious an error, since the direction is determined by considering the average spread direction over a 4 km^2 area. The model does, however, allow the eccentricity to change.

5. Flow Chart

See Figure 18.

Figure 18 Calculate Initial Perimeters

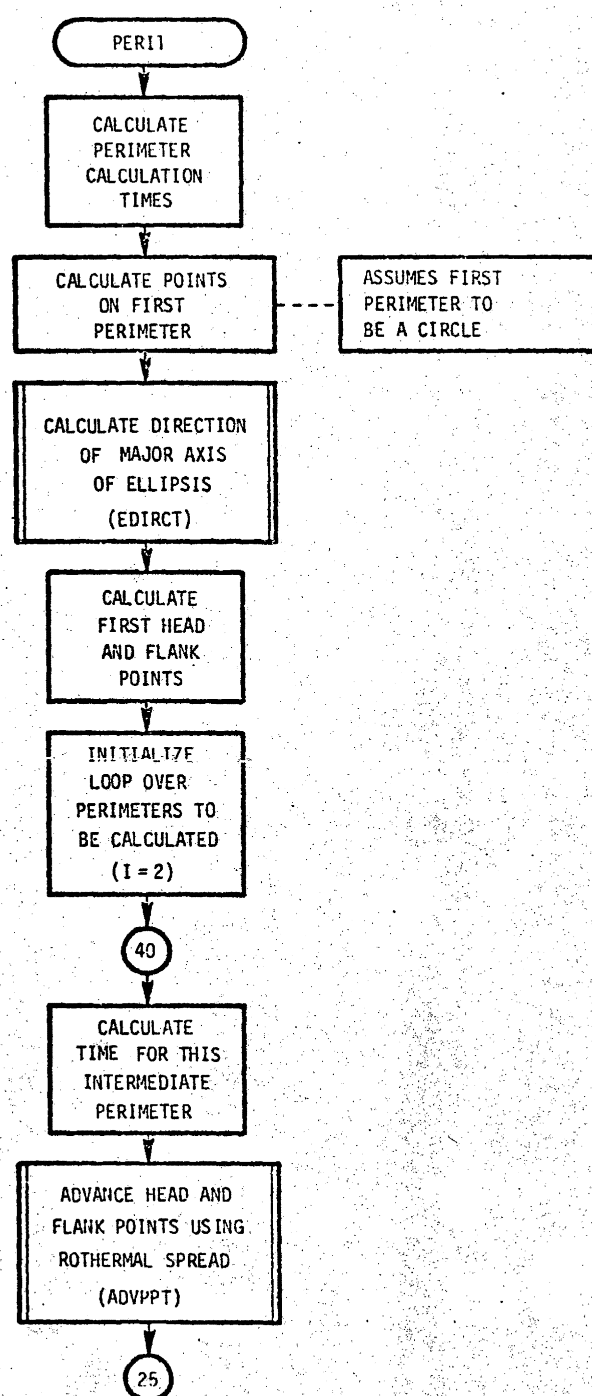
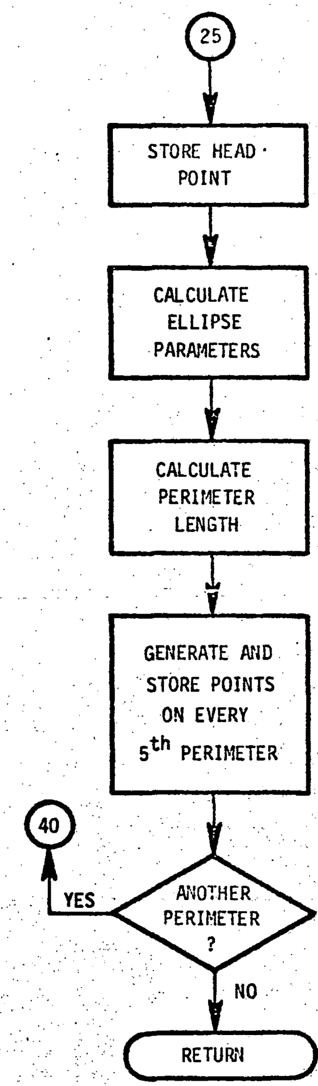


Figure 18 Calculate Initial Perimeters (Cont'd)



```
$$$$$PERI1
      $ LINK UFILE3
      SUBROUTINE FILE3
C     PRINT ,"FILE 3 NOW LOADED"
      RETURN
      END
      OPTION NOCHECK,NOLINE
      SUBROUTINE PERI1(JYOF)
C
C     THIS SUBROUTINE COMPUTES INITIAL PERIMETERS FOR THE INITIAL
C     ATTACK EVALUATION  COMMAND.
C     THE CURRENT VERSION USES AN ELLIPTIC SPREAD RATE MODEL
C     VERSION 1.20
C
C     PROGRAMMER - JANE KEEFER
C
C         INPUTS
C     TOF      - TIME OF DAY THE FIRE IS REPORTED(IN MINUTES)
C     JDOF     - DAY OF YEAR OF TOF
C     JYOF     - YEAR(YEARS-1900)
C     XF,YF    - FIRE IGNITION POINT COORDINATES(METERS)
C     AREA1    - INITIAL ESTIMATE OF FIRE SIZE(SQUARE METERS)
C     NPTRQ    - DEFAULT NO. OF PERIMETERS
C     ISPRED   - SPREAD MODEL FLAG - HARDWIRED FOR ISPRED = 1,ELLIPTIC
C     NXY      - NO. OF X,Y POINTS PER PERIMETER(MAX. ALLOWED IS 20)
C         INPUTS FROM COMMON /IMPPER/
C     TINCR    - FINE PERIMETER TIME STEP (*5=STORED PERIMETER
C                TIME INCREMENT)
C         OUTPUTS
C     ANGF     - AZIMUTH ANGLE FOR DIRECTION OF FIRE IN ELLIPTIC MODEL(ANGA,A
C     PT       - AN ARRAY CONTAINING THE PERIMETER TIMES(IN MINUTES)
C     PX       - AN ARRAY CONTAINING THE X COORDINATES OF THE PERIMETER
C     PY       - AN ARRAY CONTAINING THE Y COORDINATES OF THE PERIMETER
C     NPT      - NO. OF PERIMETER TIMES(MAX. ALLOWED IS 10)
C         OUTPUTS TO COMMON /IMPPER/
C     NTSTEP   - NUMBER OF FINE TIME STEPS (MAX=51)
C     PL       - AN ARRAY OF PERIMETER LENGTHS AT FINE TIME STEPS
C     XCHD     - AN ARRAY OF X COORDINATES OF THE FIRE HEAD
C                AT FINE TIME STEPS (EQUIVALENT TO PCNR SPACE)
C     YCHD     - AN ARRAY OF Y COORDINATES OF THE FIRE HEAD
C                AT FINE TIME STEPS (EQUIVALENT TO PCNT SPACE)
C
C         ROUTINES USED
C     ADVPPT,PPOINT,ERRREQ,EDIRCT,STEPSM
C
      COMMON /FIRE1 / XF,YF,AREA1,PTIN(11),NPTRQ,PTCAL(11),NPTCAL,IDISP
      COMMON /FMOD1 / CMESH,ISPRED,ISPREP,METIN,CTHRSH
      COMMON /TIME1 / JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
      COMMON /PERIM1/ NPT,NXY,DELT,PT(11),PX(20,11),PY(20,11)
      COMMON /CNSTNT/ RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
      COMMON /IMPPER/ TINCR,NTSTEP,PL(51),WD(51),PCNR(51),PCNT(51),
     &                PGR(51),CWR(51)
C
      DIMENSION XP(20),YP(20)
      DIMENSION XCHD(51),YCHD(51)
      EQUIVALENCE (XCHD(1),PCNR(1)),(YCHD(1),PCNT(1))
C
C
C     FILL PERIMETER TIME ARRAY
C
```

```
      NPT=MIN0(10,NPTRQ)
      NTSTEP=NPT*5+1
      DO 10 I = 1,NPT
      PT(I) = FLOAT(I-1)*TINCR*5.0
   10 CONTINUE
C
C     DEFINE SPREAD MODEL - HARDWIRED FOR ELLIPTIC
C     MODEL ONLY
C
      ISPRED = 1
C
C     COMPUTE FIRST PERIMETER RADIUS - ASSUME A CIRCLE ABOUT XF,YF
C
      RAD = SQRT(AREA1/PI)
C          COMPUTE INITIAL PERIMETER POINTS
      ANGLE = TWOPI/NXY
      DO 20 J = 1,NXY
      THETA = ANGLE*FLOAT(J-1)
      PX(J,1) = XF + RAD*COS(THETA)
      PY(J,1) = YF + RAD*SIN(THETA)
   20 CONTINUE
C
C     ADVANCE PERIMETERS  TO REMAINING TIMES
C          ELLIPTIC MODEL - ESTABLISH THE ELLIPSE ORIENTATION
C
      CALL EDIRCT(XF,YF,TOF,JDOF,JYOF,CMESH,ANGF)
C          SET INITIAL RADIAL DIRECTIONS AND X,Y POINTS
 1000 FORMAT (/6X,41HINITIAL ATTACK AZIMUTHAL FIRE DIRECTION = ,
     &   F6.2, 8H DEGREES //)
      ANGA = ANGF
      ANGB = ANGF + HALFPI
      ANGC=ANGF-HALFPI
      XAT = XF + RAD*SIN(ANGA)
      YAT = YF + RAD*COS(ANGA)
C          SAVE HEAD POINT
      XCHD(1)=XAT
      YCHD(1)=YAT
      XBT = XF + RAD*SIN(ANGB)
      XCT=XF+RAD*SIN(ANGC)
      YBT = YF + RAD*COS(ANGB)
      YCT=YF+RAD*COS(ANGC)
C          COMPUTE PERIMETERS FOR REMAINING TIMES
C
      PL(1)=PI*RAD*2.0
      TDEL=TINCR
      DO 40 I = 2,NTSTEP
      TLAST=FLOAT(I-1)*TINCR
      CALL CNRTST(TOF,JDOF,JYOF,TLAST,TOP,JDOP,JYOP)
      CALL ADVPPT(1,CMESH,ANGA,TDEL,TOP,JDOP,JYOP,XAT,YAT)
      CALL ADVPPT(1,CMESH,ANGB,TDEL,TOP,JDOP,JYOP,XBT,YBT)
      CALL ADVPPT(1,CMESH,ANGC,TDEL,TOP,JDOP,JYOP,XCT,YCT)
```

```
C        SAVE HEAD POINT
      XCHD(I)=XAT
      YCHD(I)=YAT
C        COMPUTE POINTS ON AN ELLIPSE
      XDA = XAT - XF
      YDA = YAT - YF
      XDB=AMAX1(XBT-XF,XCT-XF)
      YDB=AMAX1(YBT-YF,YCT-YF)
      RA = SQRT(XDA*XDA + YDA*YDA)
      RB = SQRT(XDB*XDB + YDB*YDB)
      A = RA*RA/(2.0*RA - RB)
      B = RA*(2.0*A - RA)
      B=SQRT(ABS(B))
      PL(I)=TWOPI*SQRT((A*A+B*B)/2.0)
      IF(MOD(I-1,5).NE.0)GO TO 35
      IPOS=(I-1)/5+1
      CALL PPOINT(A,B,ANGA,XF,YF,NXY,XP,YP)
C
      DO 30 J = 1,NXY
      PX(J,IPOS) = XP(J)
      PY(J,IPOS) = YP(J)
   30 CONTINUE
   35 CONTINUE
   40 CONTINUE
C
      RETURN
      END
```

20.0 ELLIPSE DIRECTION (EDIRCT)

1. Purpose

This subroutine establishes the direction of propagation for elliptic model perimeters.

2. Arguments

INPUTS

XF,YF	-	the coordinates of the fire ignition point
TOP	-	time of day fire reported (min. from midnight)
JDOF	-	day of year fire reported (days from 31 Dec.)
CMESH	-	sampling length or data cell size (meters)

OUTPUTS

ANGF	-	azimuth angle of propagation of elliptic model perimeters

3. Procedure

A fire spreading at a rate of about 70 feet/min. will cover a distance of approximately 1300 meters in one hour. The area covered by the fire in one hour is approximated by a 1300 by 1300 meter square with the fire ignition point (XF,YF) at the center. This routine first establishes the maximum and minimum X,Y values for such a "one-hour" fire sampling area. It then clips this area if part of it falls outside the single quadrant of prototype data that is presently available in the test data base. Later versions would not, of course, have this restriction.

Starting at the southwest corner of the sampling area, the direction and magnitude of the maximum spread rate are computed at each X,Y point separated by a distance CMESH (the data cell size) within the area. Each calculated spread rate is resolved into x and y components (V_x and V_y), and the sum

of each component is accumulated. After all the points within the area have been sampled, the average azimuthal direction of propagation is computed from the cumulative sum of the components; i.e.,

$$\alpha_f = \arctan(\Sigma V_x / \Sigma V_y) \tag{1}$$

4. COMMENTS

This routine is needed to establish an orientation for the elliptic spread model; it will not be used by the other spread models.

5. Flow Chart

See Figure 19.

of each component is accumulated. After all the points within the area have been sampled, the average azimuthal direction of propagation is computed from the cumulative sum of the components; i.e.,

$$\alpha_f = \arctan(\Sigma V_x/\Sigma V_y) \tag{1}$$

4. COMMENTS

This routine is needed to establish an orientation for the elliptic spread model; it will not be used by the other spread models.

5. Flow Chart

See Figure 19.

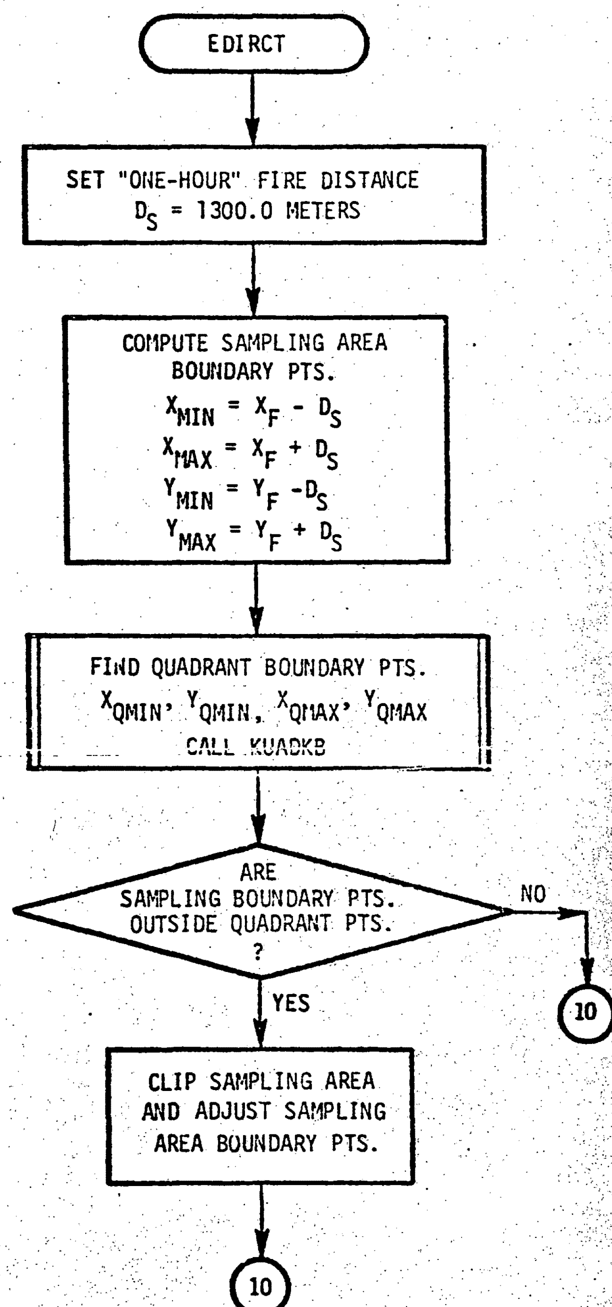


Figure 19 Logic Structure for EDIRCT

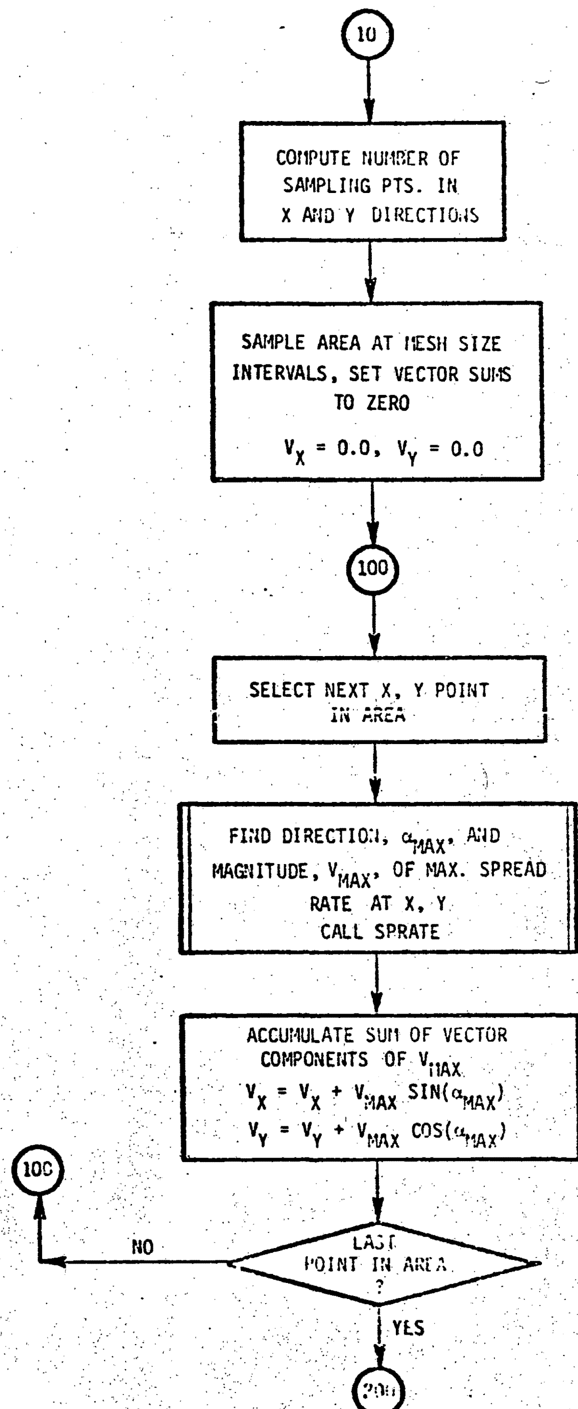


Figure 19 Logic Structure for EDIRCT (Cont'd)

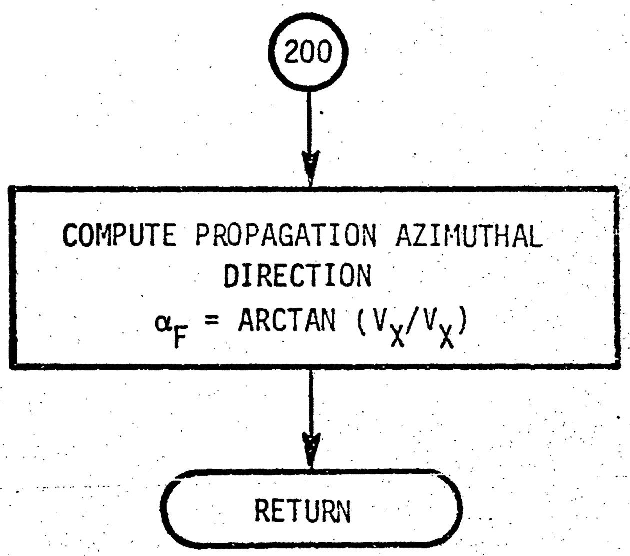


Figure 19 Logic Structure for EDIRCT (Cont'd)

```
      SUBROUTINE EDIRCT(XF,YF,TOF,JDOF,IYR,CMESH,ANGF)
C
C     THIS SUBROUTINE ESTABLISHES THE DIRECTION OF A FIRE FOR THE
C     ELLIPTIC SPREAD MODEL
C     VERSION 1.00
C
C     PROGRAMMER - J. KEEFER
C
C          INPUTS
C     XF,YF   - COORDINATES OF FIRE IGNITION POINT
C     TOF     - TIME OF DAY OF FIRE IGNITION(IN MINUTES)
C     JDOF    - DAY OF YEAR OF TOF
C     IYR     - YEAR OF TOF (YEARS - 1900)
C     CMESH   - SAMPLING LENGTH(DATA CELL SIZE)
C          INPUTS FROM COMMON /FUEL2/
C     XOF     - MINIMUM X VALUE
C     XMF     - MAXIMUM X VALUE
C     YOF     - MINIMUM Y VALUE
C     YMF     - MAXIMUM Y VALUE
C          OUTPUT
C     ANGF    - AZIMUTH ANGLE OF THE MAJOR AXIS OF THE ELLIPSE
C
C          ROUTINES USED
C     KUADKB,SPRATE,ATAN2
      COMMON/FUEL2/KUADF,XOF,YOF,XMF,YMF,DXFI,DXFII,DYFI,DYFII,
     &               DXF02,DYF02,NXFC,NYFC,NCELF,AGET(50,50),IFTYPE
C
C     ESTABLISH SAMPLING BOUNDARY FOR ONE-HOUR FIRE
C
      DSAMP = 400.0
      XMIN = XF - DSAMP
      XMAX = XF + DSAMP
      YMIN = YF - DSAMP
      YMAX = YF + DSAMP
C
C     FIND QUANDRANT BOUNDARY - HARDWIRED FOR A SINGLE QUADRANT
C
C               ASSUME QUADRANT BOUNDS SAME AS FUELS
      XQMIN=XOF
      XQMAX=XMF
      YQMIN=YOF
      YQMAX=YMF
C          CLIP AREA OUTSIDE QUADRANT - NEC. FOR SINGLE QUADRANT ONLY
      CT = CMESH*0.51
      IF(XMIN.LT.XQMIN) XMIN = XQMIN + CT
      IF(YMIN.LT.YQMIN) YMIN = YQMIN + CT
      IF(XMAX.GT.XQMAX) XMAX = XQMAX - CT
      IF(YMAX.GT.YQMAX) YMAX = YQMAX - CT
C
C     COMPUTE NUMBER OF SAMPLING POINTS ALONG BOUNDARY LENGTHS
C
      NSAMPX = IFIX((XMAX-XMIN)/CMESH)
      NSAMPY = IFIX((YMAX-YMIN)/CMESH)
      XS = XMIN - CMESH
      SUMRXS = 0.0
      SUMRYS = 0.0
```

```
C
C     SAMPLE AREA AT CMESH INTERVALS
C
      DO 100 IX = 1,NSAMPX
      XS = XS + CMESH
      YS = YMIN - CMESH
C
      DO 80 IY = 1,NSAMPY
      YS = YS + CMESH
      CALL SPRATE(XS,YS,TOF,JDOF,IYR,ANGR,ROS,ROSMAX,ANGMAX)
      RTX = ROSMAX*SIN(ANGMAX)
      RTY = ROSMAX*COS(ANGMAX)
      SUMRXS = SUMRXS + RTX
      SUMRYS = SUMRYS + RTY
   80 CONTINUE
C
  100 CONTINUE
C
C     COMPUTE ANGLE FROM VECTOR SUM OF X AND Y RATE COMPONENTS
C
      ANGF = ATAN2(SUMRXS,SUMRYS)
C
      RETURN
      END
```

21.0 SPREAD RATE CALCULATION (SRPATE)

1. Purpose

This subroutine calculates the spread rate at a point in any direction. It also calculates the direction and magnitude of the maximum spread rate at the point.

2. Arguments

<u>INPUTS</u>

X,Y	-	the coordinates of the point for which the spread rate is being computed.
TIME	-	time of day (minutes from midnight)
IDATE	-	day of year (in days from 31 December)
ALPHA	-	azimuthal direction (radians) for which spread rate is to be calculated.

<u>OUTPUTS</u>

VALPHA	-	the spread rate (meter/minute) in the direction, VALPHA.
VMAX	-	the maximum spread rate (meter/min.)
ALPMAX	-	the azimuthal direction of VMAX (radians)

3. Procedure

This routine gathers the necessary data on wind, slope, and their directions together with fuel data. These are processed, and a fuel model is established. The maximum spread rate and its direction, together with the spread rate in any given direction are computed using the equations described below.

Vector spread rate equations:

Given: V_o = no slope, no wind spread rate,

$\vec{\phi}_s$ = dimensionless slope factor affecting V_o,

$\vec{\phi}_w$ = dimensionless wind factor affecting V_o,

α_s = azimuthal direction of downward slope at x,y,

α_w = azimuthal direction of wind at x,y,

a vector diagram can be constructed as shown in Figure 20.

The vectors $\vec{\phi}_w$ and $\vec{\phi}_s$ are added to obtain the vector $\vec{\phi}_{ws}$ which represents the combined effect of wind and slope on the no-slope, no-wind spread rate, V_o. The magnitude of this vector is then added to scaler spread rate, V_o. The wind-effect and slope-effect components, f_E, and f_N, of ϕ_{ws} in the direction of α_{max} are given by

$$f_E = f_{se} + f_{wE} = \phi_w \sin\alpha_w + \phi_s \sin\alpha_s, \quad (1)$$

$$f_N = f_{sN} + f_{wN} = \phi_w \cos\alpha_w + \phi_s \cos\alpha_s \quad (2)$$

Define

$$\phi_{ws} = \sqrt{f_E^2 + f_N^2} = \text{magnitude of } \vec{\phi}_{ws}, \quad (3)$$

and

$$V_{max} = V_o(1 + \phi_{ws}) = \text{magnitude of } \vec{V}_{max}. \quad (4)$$

The azimuthal angle of V_{max} is, defined by (see Figure 21)

$$\alpha_{max} = \arctan(f_E/f_N). \quad (5)$$

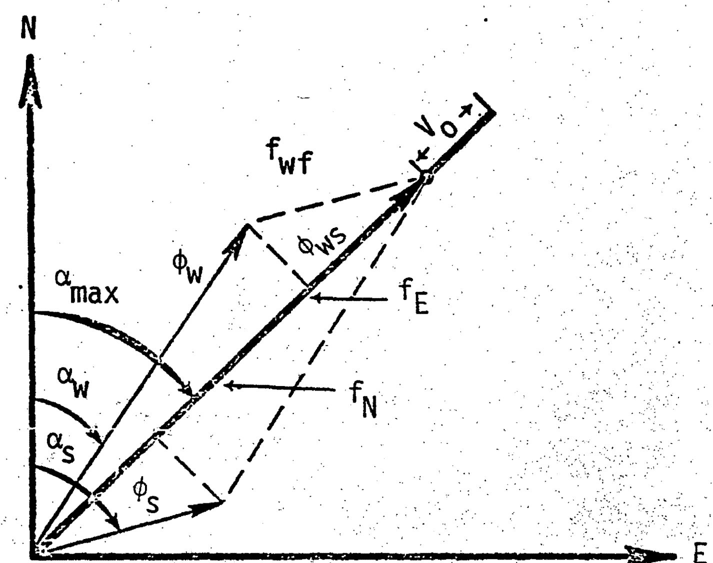


Figure 20 Vector diagram for elliptic spread model. See text for explanation of symbols

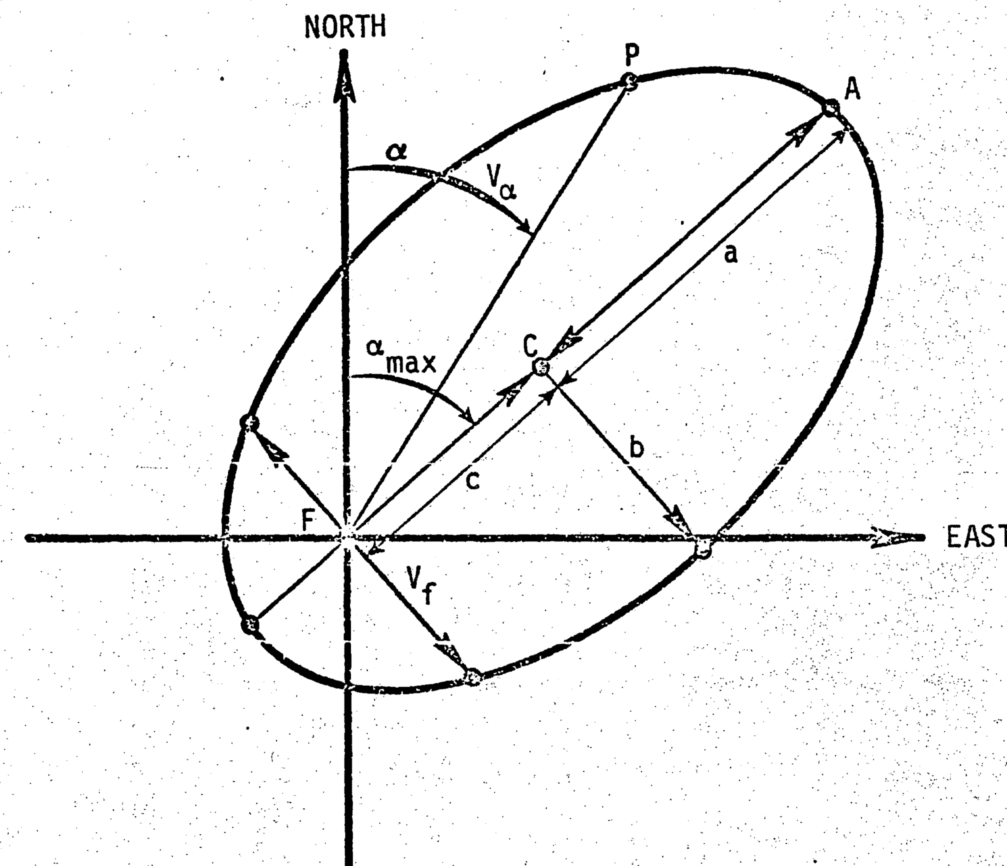


Figure 21. Elliptic spread rate model diagram

For an elliptical spread model, the flank spread rate $\vec{V}_f$ is required (i.e., the spread rate normal to $\vec{V}_{max}$). This is obtained by finding the projections of $\vec{\phi}_w$ and $\vec{\phi}_s$ that are normal to $\vec{V}_{max}$.

From Figure 20, the magnitude of the normal projections is given by

$$f_{wf} = \phi_w \; |\sin(\alpha_{max} - \alpha_w)| = \text{flank wind factor for } V_o \qquad (6)$$

$$f_{sf} = \phi_s \; |\sin(\alpha_{max} - \alpha_s)| = \text{flank slope factor for } V_o \qquad (7)$$

Since α_s is in the direction of up-slope, f_{wf} and f_{st} are in the same direction, and

$$\vec{V}_f = V_o(1 + |f_{wf} + f_{sf}|) \; \hat{a}_f. \qquad (8)$$

Figure 21 shows the relation between the computed quantities and the elliptic spread rate model. The point x,y is located at the focus F, and the center of the ellipse is at point C. The eccentricity of the ellipse in Figure 21 is given by

$$e = 1 - \frac{V_f}{V_{max}} = \frac{c}{a}, \qquad (9)$$

but

$$V_{max} = a + c \qquad (10)$$

thus,

$$a = \frac{V_{max}}{1+e}. \qquad (11)$$

The spread rate, V_α, in an arbitrary azimuthal direction α is given by

$$V_\alpha = \frac{a(1 - e^2)}{1+e \cos\theta}, \tag{12}$$

where

$$\theta = \alpha_{max} - \alpha + \pi$$

In programming these equations, the aspect azimuth (α_a) returned by the TOPO routine is in the direction of the downward slope. The angle α_s in Figure 1 is related to α_a by

$$\alpha_s = \alpha_a - \pi \tag{13}$$

This accounts for the sign change of the trigonometric functions associated with α_s in the coding, as the program uses the α_a variable.

A further note on angles. Any polar angle (ϕ) is related to the azimuth angle by

$$\phi = \frac{\pi}{2} - \alpha. \tag{14}$$

Thus, use of an azimuth angle to compute x,y coordinates instead of the usual polar angle amounts to interchanging the x,y axes.

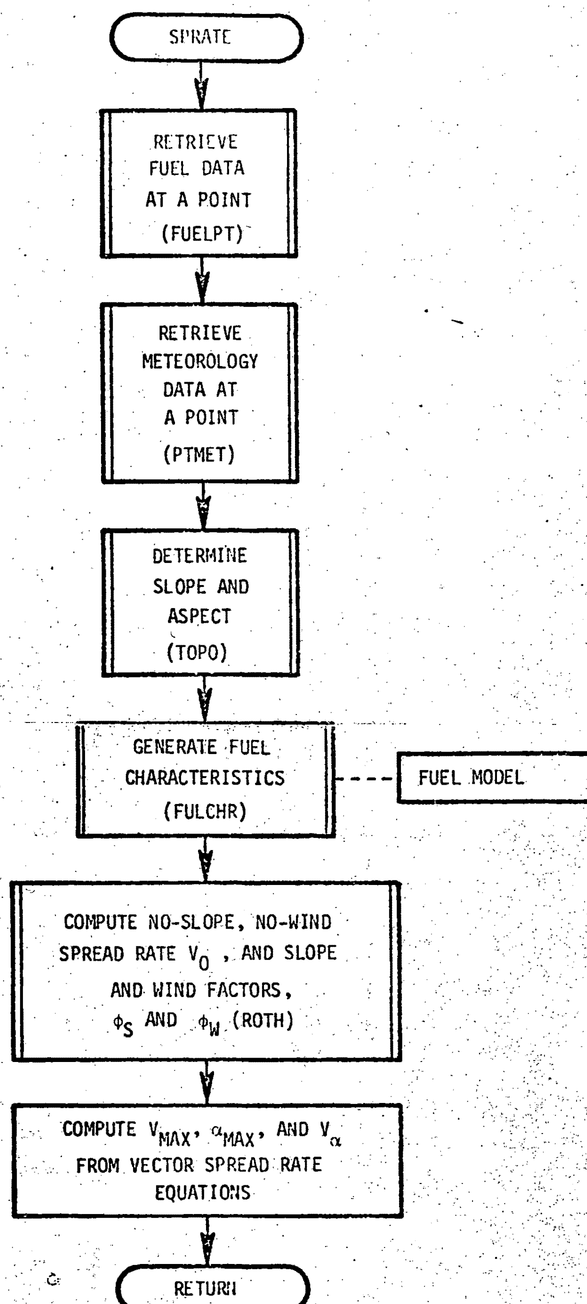
4. COMMENTS

The directional generalization of spread rate presented here is very preliminary, later versions will certainly include changes resulting from comparisons with real-world directional spread rate characteristics.

5. Flow Chart

See Figure 22.

Figure 22 Spread Rate



```
      SUBROUTINE SPRATE(X,Y,TIME,IDATE,IYR,ALPHA,VALPHA,VMAX,ALPMAX)
C
C     THIS SUBROUTINE CALCULATES THE SPREAD RATE AT A POINT IN ANY
C     DIRECTION.  IT ALSO CALCULATES THE DIRECTION AND MAGNITUDE OF
C     MAXIMUM SPREAD RATE AT THAT POINT.
C     VER 2.0
C
C     VECTOR FORM OF ROTHERMAL EQUATION - DR. JAMES C. SANDERLIN
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C          INPUTS
C     X       - X COORDINATE OF LOCATION
C     Y       - Y COORDINATE OF LOCATION
C     TIME    - TIME OF DAY (MINUTES)
C     IDATE   - DAY OF YEAR
C     IYR     - YEAR (YEARS - 1900)
C     ALPHA   - AZIMUTHAL ANGLE OF INTEREST (RADIANS)
C          OUTPUTS
C     VALPHA - RATE OF SPREAD IN DERECTION ALPHA
C     VMAX   - MAXIMUM RATE OF SPREAD
C     ALPMAX - AZIMUTHAL ANGLE OF MAXIMUM SPREAD RATE
C
C          ROUTINES USED
C     FUELPT,PTMET,FHUM,TOPO,FULMOD,ROTH
C
      COMMON /BINEW / WO(2,6),SIG(2,6),ST(2,6),SE(2,6),H(2,6),AMF(2,6),
     & RHOP(2,6),AMX(2,6),DEPTH,SLOPE,WIND,M,N(2)
      DIMENSION ERRR(17)
      DATA PI/3.141592/
      DATA TWOPI/6.283184/
      DATA AR450/7.85398/
      DATA AR270/4.712389/
      DATA HALFPI/1.570796/
      DATA ERRR/1.0,0.85,0.7,
     &  0.6857,0.6714,0.6571,0.6429,0.6286,           0.6143,
     & 0.6000, 0.5857,0.5714,0.5571,0.5429,0.5286,0.5143,0.5/
C
C     GET FUEL DATA AT THIS POINT
C
      CALL FUELPT(X,Y,ICLASS,VALUE,AGE,IFTYPE)
C
C     GET METEORLOGICAL DATA AT THIS POINT AND TIME
C
      CALL PTMET(X,Y,TIME,IDATE,WAS,WVEL,SDWAS,SDWVEL,
     & RELHUM,TEMP)
      WIND=WVEL*3.2808
C
C     CALCULATE ASPECT AZIMUTH AT POINT AND SLOPE IN DERECTION OF ASPECT
C
      CALL TOPO(4,X,Y,0.0,ZOUT,SRUF,SLE,SLN,ASPECT,SLP)
      SLOPE=SLP
C
C     USE A FUEL MODEL TO GENERATE ROTHERMAL INPUTS
C
      CALL FULCHK(IYR,IDATE,TIME,
     & TEMP,RELHUM,WVEL,IFTYPE,AGE)
C
C     CALCULATE VO,PHIS,PHIW
C
      CALL ROTH(VO,PHIS,PHIW)
C
```

```
C     CALCULATE DESIRED OUTPUTS USING A VECTOR FORM OF ROTHERMAL
C     EQUATION BY SANDERLIN
C
      FE=PHIW*SIN(WAS)-PHIS*SIN(ASPECT)
      FN=PHIW*COS(WAS)-PHIS*COS(ASPECT)
      VMAX=V0*(1.0+SQRT(FE*FE+FN*FN))
      IF(VMAX.LE.1.0E-37)GO TO 30
      IF(FE.EQ.0.0.AND.FN.EQ.0.0)GO TO 30
      ALPMAX=HALFPI-ATAN2(FN,FE)
      VF=V0*(1.0+ABS(PHIW*SIN(ALPMAX-WAS)-PHIS*SIN(ALPMAX-ASPECT)))
C
C     CALCULATE E AND A FOR SPREAD ELLIPSE
C
      E=1.0-VF/VMAX
      INDEX=IFIX(WVEL*0.007455)+1
      EPRIME=E*AMIN1(ERRR(INDEX)*1.3,1.0)
      A=VMAX/(1.0+EPRIME)
C
      THETA=ALPHA-PI-ALPMAX
      IF(THETA.GT.PI)THETA=AMOD(THETA,TWOPI)
      IF(THETA.GT.PI)THETA=THETA-TWOPI
      IF(THETA.LT.-PI)THETA=AMOD(THETA,-TWOPI)
      IF(THETA.LT.-PI)THETA=THETA+TWOPI
      TURB=1.0
C          INCLUDE EFFECTS OF TURBULANCE
   50 VALPHA=TURB*A*(1.0-EPRIME*EPRIME)/(1.0+EPRIME*COS(THETA))
C
   20 RETURN
   30 ALPMAX=ALPHA
      VALPHA=V0
      GO TO 20
      END
```

22.0 ADVANCE PERIMETER POINT (ADVPPT)

1. Purpose

This routine advances a perimeter point for a specified time increment.

2. Arguments

Input

KOPT	-	Output option KOPT = 1, Advance the perimeter point in the direction ANGR KOPT = 2, Advance the perimeter point in the direction of the maximum spread rate
CMESH	-	Sampling length (meters) or Data Cell size.
ANGR	-	Azimuthal direction of advancement needed by option 1.
TDEL	-	Time increment for advancement of XP, YP (in minutes)
TOP	-	Simulation time of day of perimeter point XP,YP (in minutes from midnight)
JDOP	-	Simulation day of year of TOP

Input/Output

XP,YP	-	The perimeter point coordinates being advanced. On input, XP,YP is the perimeter point at TOP. On output, XP,YP is the perimeter point at TOP + TDEL.

3. Procedure

The input coordinates XP, YP are stored with the time increment. Depending upon the option selected, the appropriate spread rate at XP,YP is obtained by a call to subroutine SPRATE. This rate is used to compute the

time (TAU) required to cover a distance equal to the data cell size, CMESH. If this time is less than the specified time increment, a new x,y point is established from the above rate and the appropriate angle. The time increment is reduced by TAU and the above procedure is repeated until TAU exceeds the specified time increment. The final rate is used to compute the appropriate output coordinates.

4. Alternatives

The current version of this routine is basically independent of any particular spread model. The KOPT = 1 option can be selected for elliptic or fixed radii, radial spread models, while the KOPT = 2 option makes the routine suitable for a segmented spread model or a cell type spread model.

At present, the input time of day, TOP, is not incremented by TAU, although in principle it should be. However, the dependence of the rate of spread on the time of day is mostly through the meteorology data, and this is not expected to change for most time increments encountered within this routine.

5. Flow Chart

See Figure 23.

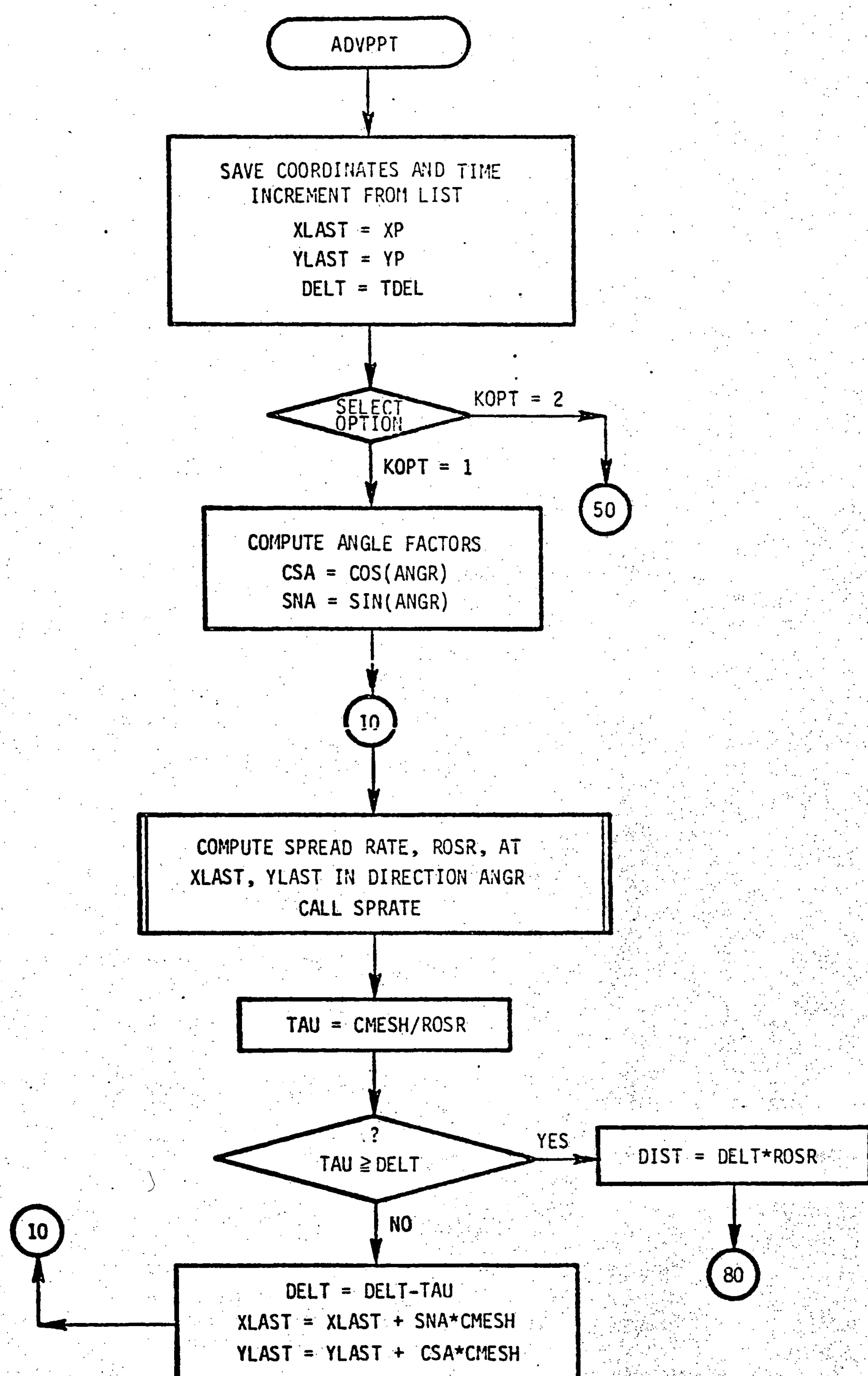


Figure 23 Logic Structure for Routine ADVPPT

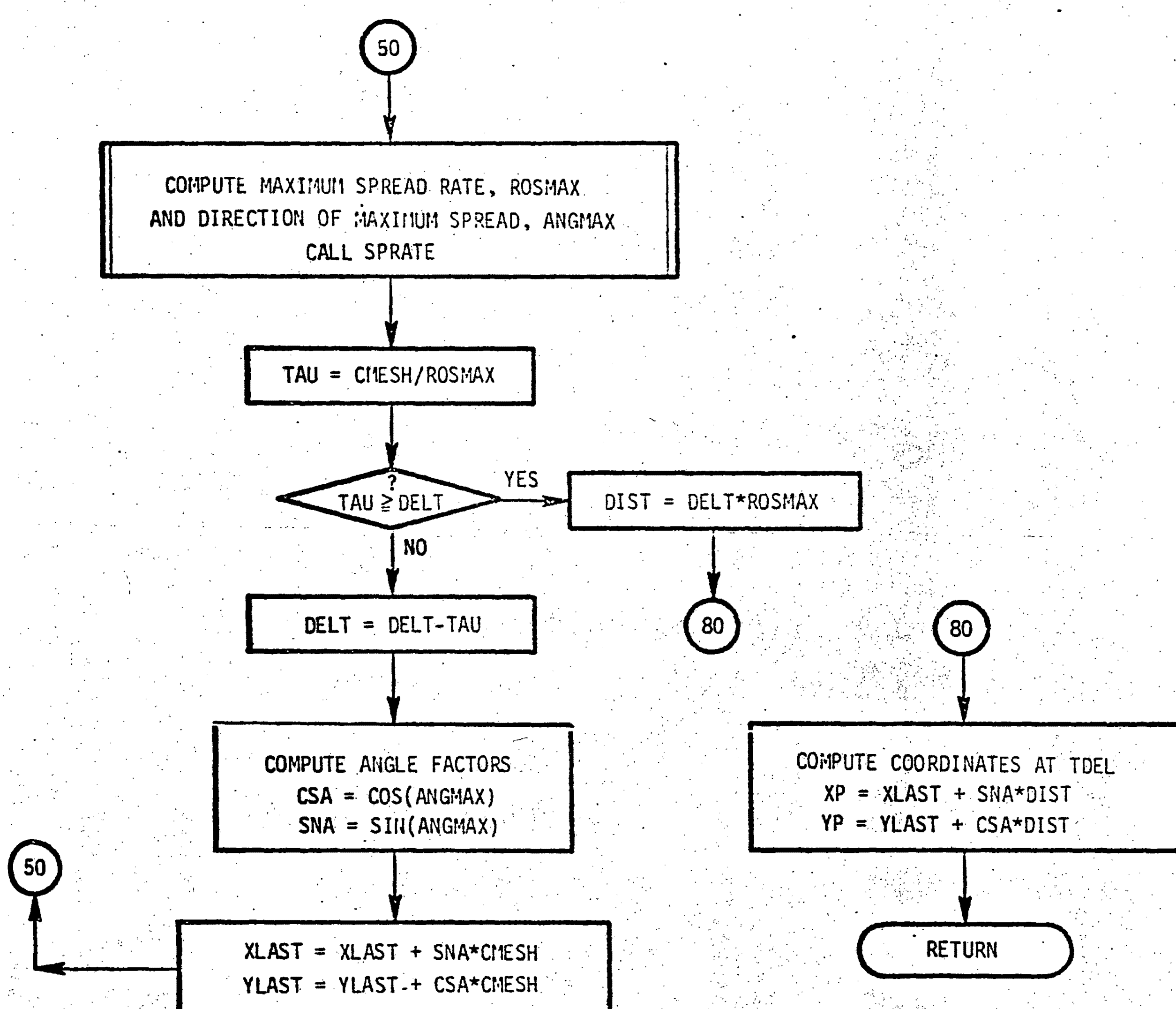


Figure 23 Logic Structure for Routine ADVPPT (Cont'd)

```
      SUBROUTINE ADVPPT(KOPT,CMESH,ANGR,TDEL,TOP,JDOP,IYR,XP,YP)
C
C     THIS SUBROUTINE ADVANCES A PERIMETER POINT
C     VERSION 1.00
C
C     PROGRAMMER - J. KEEFER
C
C        INPUTS
C     KOPT     - OUTPUT OPTION
C                KOPT = 1, ADVANCE XP,YP IN DIRECTION OF ANGR
C                KOPT = 2, ADVANCE XP,YP IN DIRECTION OF MAX. SPREAD RATE
C     CMESH    - SAMPLING LENGTH(DATA CELL SIZE)
C     ANGR     - AZIMUTHAL DIRECTION OF ADVANCEMENT NEEDED BY OPTION 1
C     TDEL     - TIME INCREMENT FOR ADVANCEMENT OF XP,YP
C     TOP      - SIMULATION TIME OF DAY REPRESENTED BY XP,YP
C     JDOP     - SIMULATION DAY OF YEAR FOR TOP
C     IYR      - YEAR FOR JDOP (YEARS - 1900)
C        INPUT/OUTPUT
C     XP,YP    - PERIMETER POINT BEING ADVANCED
C                ON INPUT, XP,YP IS POINT AT TOP
C                ON OUTPUT, XP,YP IS POINT AT TOP + TDEL
C
C        ROUTINES USED
C     SPRATE,STEPSM
C
C     SET LAST POINT AND SAVE TDEL
C
      XLAST = XP
      YLAST = YP
      DELT = TDEL
C        SELECT OPTION
      IF(KOPT.EQ.2) GO TO 50
C
C     OPTION 1 - ADVANCE PERIMETER PT. IN DIRECTION ANGR
C
      CSA = COS(ANGR)
      SNA = SIN(ANGR)
C
   10 CONTINUE
      CALL SPRATE(XLAST,YLAST,TOP,JDOP,IYR,ANGR,ROSR,ROSMAX,ANGMAX)
      TAU = CMESH/ROSR
      IF(TAU.GE.DELT) GO TO 20
C
      DELT = DELT - TAU
      XLAST = XLAST + CMESH*SNA
      YLAST = YLAST + CMESH*CSA
      GO TO 10
C
   20 CONTINUE
      DIST = DELT*ROSR
      GO TO 80
C
C     OPTION 2 - ADVANCE PERIMETER POINT IN DIRECTION OF MAX. SPREAD
C
   50 CONTINUE
      CALL SPRATE(XLAST,YLAST,TOP,JDOP,IYR,ANGR,ROSR,ROSMAX,ANGMAX)
      TAU = CMESH/ROSMAX
      IF(TAU.GE.DELT) GO TO 60
      DELT = DELT - TAU
      CSA = COS(ANGMAX)
      SNA = SIN(ANGMAX)
      XLAST = XLAST + CMESH*SNA
      YLAST = YLAST + CMESH*CSA
      GO TO 50
C
   60 CONTINUE
      DIST = DELT*ROSMAX
C
   80 CONTINUE
      XP = XLAST + DIST*SNA
      YP = YLAST + DIST*CSA
C
      RETURN
      END
```

23.0 ELLIPTICAL PERIMETER POINT ROUTINE (PPOINT)

1. Purpose

This subroutine computes x,y points on the perimeter of an ellipse.

2. Arguments

<u>Input</u>

A	-	the semi-major axis length of the ellipse (meters)
B	-	the semi-minor axis length of the ellipse (meters)
ANGA	-	the azimuthal angle of the semi-major axis (radians)
XF,YF	-	x,y coordinates of the fire ignition point
NXY	-	the number of points on the perimeter to be calculated.

<u>Output</u>

XP,YP	-	the arrays containing the x,y coordinates of the perimeter points calculated

3. Procedure

The elliptic spread model contains a semi-major and a semi-minor axis, a and b at a given time. The angle of orientation of the ellipse is assumed to have been determined. The perimeter is located by setting the fire ignition point (X_f,Y_f) at the focus in the lower half of the ellpise as shown in Figure 24.

The first step is to compute coordinates of the ellipse center from

$$c = \sqrt{a^2 + b^2},$$

$$X_c = X_f + c*\cos(\alpha),$$

$$Y_c = Y_f + c*\sin(\alpha).$$

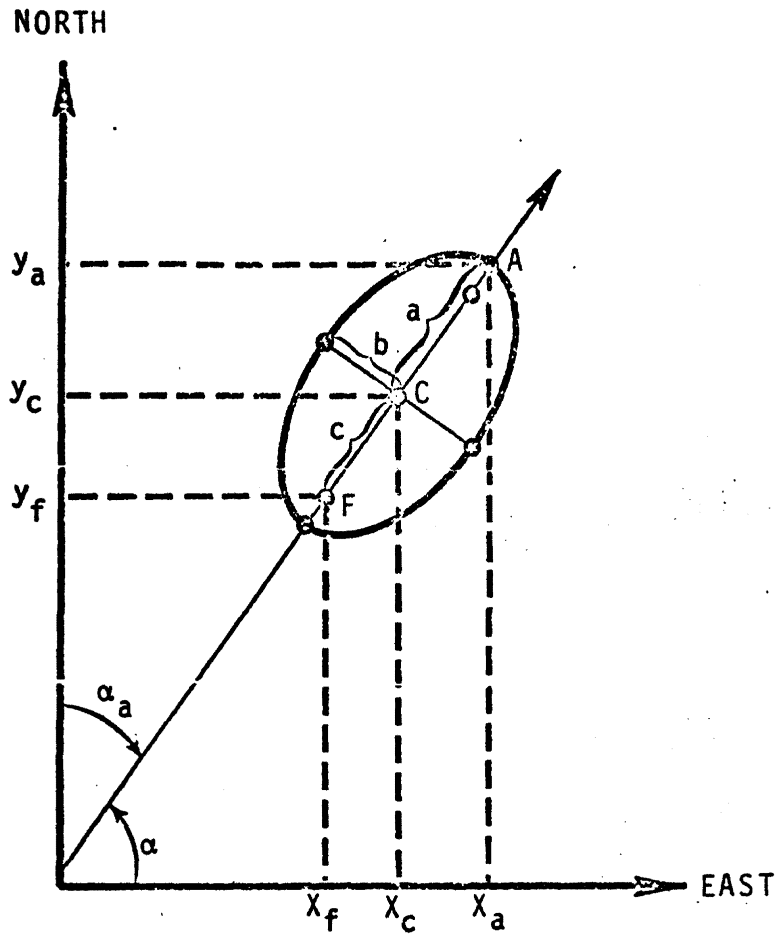


Figure 24 Elliptic perimeter diagram. Point C is the center of the ellipse, the fire ignition point is at point F and the head of the fire is located at point A.

Next, a set of NXY angles, radiating from the center coordinate, is established such that the j^{th} angle, θ, is given by

$$\theta_j = \frac{2}{N_{xy}} (j-1), \qquad j = i, N_{xy}$$

where, N_{xy} = the number of perimeter points being computed. In order to ensure that the computed points contain the end-points of the a and b major-axis. N_{xy} should be a multiple of four.

The x,y coordinates on the perimeter are then calculated relative to the center X_c, Y_c from

$$r_j = \frac{ab}{\sqrt{a^2 \sin^2\theta_j + b^2 \cos^2\theta_j}},$$

$$X_j = r_j \cos\theta_j,$$

$$Y_j = r_j \sin\theta_j.$$

The translation to the absolute coordinate system and the rotation through the angle α is accomplished by

$$X_{pj} = X_j \cos\alpha - Y_j \sin\alpha + X_c,$$

$$Y_{pj} = X_j \sin\alpha + Y_j \cos\alpha + Y_c$$

where X_{pj}, Y_{pj} are the perimeter points stored in the arrays XP,YP and returned by this routine. Thus, for example, if NXY = 12, the 12 X,Y points computed are shown in Figure 25.

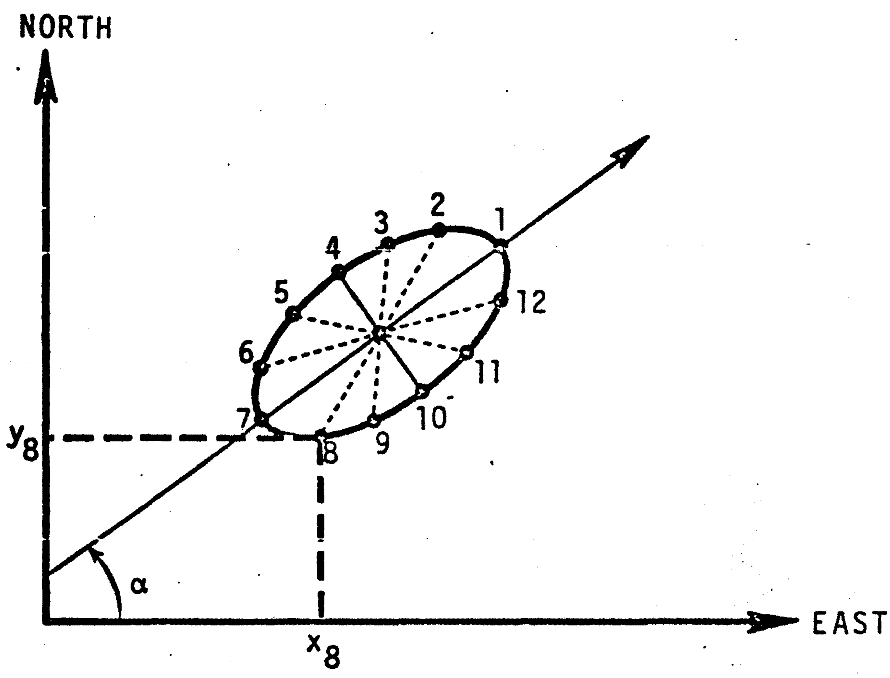


Figure 25. Elliptic perimeter diagram illustrating the order in which points are calculated and stored within the computer.

4. Comments

This routine is an explicit part of the elliptic perimeter model.

5. Flow Chart

See Figure 26.

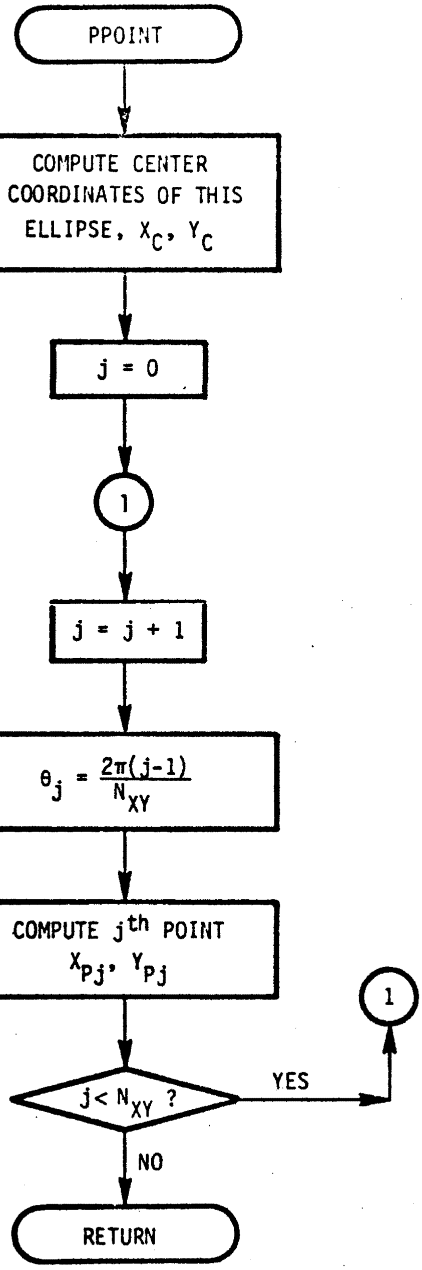


Figure 26 Logic Structure for Routine PPOINT

```
      SUBROUTINE PPOINT(A,B,ANGA,XF,YF,NXY,XP,YP)
C
C     THIS SUBROUTINE COMPUTES X,Y POINTS OF A PERIMETER ON AN ELLIPSE
C     WITH A GIVEN A,B AND ANGLE ORIENTATION
C     VERSION 1.00
C
C     PROGRAMMER - J KEEFER
C
C          INPUTS - FROM LIST
C     A         - SEMI-MAJOR AXIS(METRES)
C     B -       - SEMI-MINOR AXIS(METRES)
C     ANGA      - AZIMUTH ANGLE OF A-AXIS
C     XF,YF     - THE COORDINATES OF THE FIRE IGNITION POINT
C     NXY       - NO. OF PERIMETER POINTS TO BE CALCULATED
C          OUTPUTS
C     XP,YP     - THE COORDINATES OF THE PERIMETER POINTS
C
C          ROUTINES USED
C     SIN,COS
C
      COMMON /CNSTNT/ RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
C
      DIMENSION XP(20),YP(20)
C
C     COMPUTE THE X,Y COORDINATES OF THE CENTRE OF THE ELLIPSE
C
      ANG = HALFPI-ANGA
      CSA = COS(ANG)
      SNA = SIN(ANG)
      A2 = A*A
      B2 = B*B
      C = SQRT(ABS(A2-B2))
      XC = XF + C*CSA
      YC = YF + C*SNA
C
C     COMPUTE PERIMETER POINTS AT EQUIDISTANT ANGLES ABOUT THE CENTRE
C
      ANGLE = TWOPI/FLOAT(NXY)
      DO 20 J = 1,NXY
      THETA = ANGLE*FLOAT(J-1)
      CSJ = COS(THETA)
      SNJ = SIN(THETA)
      RVAL = A*B/SQRT(A2*SNJ*SNJ + B2*CSJ*CSJ)
      X = RVAL*CSJ
      Y = RVAL*SNJ
      XP(J) = X*CSA - Y*SNA + XC
      YP(J) = X*SNA + Y*CSA + YC
   20 CONTINUE
C
      RETURN
      END
```

24.0 FUEL CHARACTERISTICS FULCHR (IFTYP,AGE,DAY,FLOD)

Version 1.00 21 January 1976

1. PURPOSE

This subroutine provides the rate of spread model with required fuel characteristics.

2. ARGUMENTS

INPUTS

IFTYP	=	fuel type index
	=	1, short grass
	=	2, long grass
	=	3, brush (non-chaparral)
	=	4, chamise
	=	5, mixed chaparral
AGE	=	fuel age (yrs)
DAY	=	number of days since last rainfall
FLOD	=	fuel loading (gm/m^2)

OUTPUTS

(See COMMONS USED)

COMMONS USED

INPUT: None.

OUTPUT

WO(I,J)	=	fuel loading per fuel category (I) and fuel size class (J), oven-dry weight per unit area ($kg/\hat{m}^2$)
SIG(I,J)	=	Fuel particle surface-area-to-volume ratio per fuel category (I) and fuel size class (J) (M^{-1})

ST(I,J)	=	fuel particle total mineral content, mineral weight as a fraction of oven-dry fuel weight (units).
SE(I,J)	=	fuel particle silica-free mineral content, mineral weight as a fraction of oven-dry fuel weight (units).
H(I,J)	=	fuel particle low heat content per unit oven-dry weight (Joules/kg)
RHOP(I.J)	=	fuel particle density, oven-dry weight per unit volume (kg/liter)
DEPTH	=	fuel depth (meters)
M	=	fuel categories
	=	1, dead fuel
	=	2, live fuel
N(M)	=	fuel size classes per category
	=	1, leaves
	=	2, stems < 0.25 inch diameter
	=	3, 0.25 $\leq$ stems < 0.5 inch diameter
	=	4, 0.5 $\leq$ stems < 1.0 inch diameter
	=	5, 1.0 $\leq$ stems < 3.0 inches diameter
	=	6, not currently used.

3. PROCEDURE

The models of fuel characteristics are based upon a very scanty data base, however, the structure of the fuel characteristics model fulfills the requirement to fill a set of arrays to be used by a rate of spread model of the Rothermel[4] type in calculating wildland fire rates of spread. As such, the model structure is well-defined, and need not be expected to change with increasing availability information on the values of the specific fuel parameters. The model is thus structured in considerable

detail, although the actual contents are often repetitive and in some cases of very questionable authority. This model was developed with the expectation that as more information becomes available on fuel characteristics of interest it will be only necessary to change the contents of the model, while the overall structure may be expected to remain unchanged.

Three sources of data were found on fuel characteristics for fire rate of spread models of the Rothermel type, Reference 5 presents a definition of dynamic fuel models for chamise and mixed chaparral. Reference 6 presents a description of the basis for nine standardized groups of fuel descriptors that are employed as fuel models by the National Fire-Danger Rating System. Reference 3 presents representative values of fuel parameters for 11 preliminary fuel models for the National Fire-Danger Rating System.

The data extracted from these three references, and upon which the model contents are based are presented in Tables 1, 2 and 3, respectively. The model defines all of the fuel parameters required for two categories (live and dead) and up to six size classes of fuels as input data to the rate of spread model. Moisture-related fuel parameters (i.e., fuel moisture, and moisture of extinction) are omitted from this model, but are provided by separate models (FMOISD, FMOISL). Total fuel loading, also needed by several transport rate and work rate functions, is provided by a separate model (FULOD). In the following paragraphs the procedures used to obtain the characteristics of each fuel type are described.

Table 1. Fuel Characteristics Extracted From Reference 5

FUEL CHARACTERISTICS	UNITS	SYMBOL	CATEGORY, SIZE CLASS[1]	FUEL TYPE	
				(PURE) CHAMISE	MIXED CHAPARRAL
Total Fuel Loading	Tons/Acre	W_n	N/A[2]	0.0459A/(1.4459 + 0.0315A)[3]	0.0459A/(0.4849 + 0.0170A)
Fraction Dead	None	F_d	N/A	0.0694 exp (0.0402A	0.0694 exp (0.0402A)
Fractional Fuel Loading by Fuel Category and Size Class[4]	Tons/Acre	$W_{o_{i,j}}$	1,1	N/A[5]	
			1,2	$0.347\ F_d\ W_n$	
			1,3	$0.364\ F_d\ W_n$	
			1,4	$0.207\ F_d\ W_n$	
			1,5	$0.085\ F_d\ W_n$	
			2,1	$(0.1957 - 0.305\ F_d)\ W_n$	
			2,2	$(0.2416 - 0.256\ F_d)\ W_n$	
			2,3	$(0.1918 - 0.256\ F_d)\ W_n$	
			2,4	$(0.2648 - 0.050\ F_d)\ W_n$	
			2,5	$(0.1036 - 0.114\ F_d)\ W_n$	
Heat Content[6]	BTU/LB	h	2,1 & 2,2	$9613.0 - D + 0.1369D^2 + 3.65E - 4D^3$	
			2,3;2,4 & 2,5	$9509.0 - 10.74D + 0.1359D^2 - 4.05E - 4D^2$	
Fuel Depth [7]	FT^{-1}	d	N/A	$d_n/d_{50} = (4.92E - 2)h - (5.83E - 4)n^2$ $d_{50} = 4.25 + 0.157W_{50} - 9.57E - 4\ W_{50}^2$	
				$W_{50} = 16.2$	$W_{50} = 37.4$
Fuel Surface-To-Volume Ratio[8]	FT^{-1}	-	1	2200.0	
			2	640.0	
			3	127.0	
			4	61.0	
			5	27.0	
Silica-Free Mineral Content[8]	N/A	-	1	0.035	
			2	0.015	
			3	0.015	
			4	0.015	
			5	0.015	
Fuel Particle Density	LB/FT^3	-	1	32.0	
			2	46.0	
			3	46.0	
			4	46.0	
			5	46.0	

Notes:
[1] There are two fuel categories: i = 1, dead fuel; i = 2, line fuel.
[2] N/A = not applicable.
[3] A = age (yrs.)
[4] Average mortality conditions.
[5] Leaves are omitted from dead fuels.
[6] Only heat content for living fuels is explicitly specified.
[7] Equations fit to curves in Reference 1: d_n = depth at n years age; d_{50} = depth at 50 years age; W_{50} = fuel loading at 50 years age.
[8] The same for both categories.

Table 2. Fuel Characteristics Extracted from Reference 6

FUEL CHARACTERISTIC	UNITS	SYMBOL	CLASS [1]	FUEL TYPE[2] A GRASS	F YOUNG CHAMISE	D MIDDLE AGE CHAPARRAL	B MATURE CHAPARRAL	C SCRUB OAK BRUSH
Fuel Loading By Class	gm/cm^2	W_F	1	0.028	0.022	0.034	0.112	0.034
		W_{Io}	2	N/A	0.011	0.056	0.09	0.022
		W_{Iou}	3	N/A	N/A	0.045	0.045	N/A
		W_L	4	N/A	0.045	N/A	0.045	N/A
Fuel Depth	cm	N/A	N/A	23.	61.	76.	183.	30.
Fuel Surface-To-Volume Ratio	cm^{-1}	σ	1	98.4	49.2	57.4	65.6	88.5
			2		3.58			
			3	[2]	0.98			
			4		49.2			
				[3] RANGE OF VARIABILITY			VALUE USED IN NFDR FUEL MODELS	
Fuel Particle Density	gm/cm^3	ρ	ALL	0.35-0.65			0.512	
Low Heat of Combustion	cal/gm	N/A	ALL	3600-5100			4400	
Total Ash Content	Percent	N/A	ALL	0-16			5	
Silica-Free Ash Content	Percent	N/A	ALL	0-5			2	

Notes:

[1] Four classes are defined: (1) fire fuels, (2) 10-hour time log fuels, (3) 100-hour time log fuels, (4) living fuels.

[2] Fuel Type — Description

Fuel Type	Description
A	Western range grasses.....
B	Mature southwest chaparral, ...
C	..., wire grass-scrub oak, ...
D	..., chaparral that doesn't qualify for "B."
F	..., young chamise and marzanita.

[3] Values apply to all fuel types.

Table 3. Fuel Characteristics Extracted from Reference 4

FUEL CHARACTERISTIC	UNITS	SYMBOL	CLASS[1]	FUEL TYPE			
				SHORT GRASS	TALL GRASS	BRUSH	CHAPARRAL
Total Loading	Tons/Acre	N/A	N/A	0.75	3.0	6.0	25.0
Loading Per Class	LB/FT^2	W_0	1	0.034	0.138	0.046	0.230
			2	-	-	0.023	0.184
			3	-	-	-	0.092
			4	-	-	0.092	0.230
Surface-To-Volume Ratio	FT^{-1}	σ	1	3500	1500	2000	2000
			2	-	-	109	109
			3	-	-	-	30
			4	-	-	1500	1500
Fuel Depth	FT	δ	N/A	1.0	2.5	2.0	6.0
Total Mineral Content	N/A	S_T	ALL	[2] 0.0555			
Silica-Free Mineral Content	N/A	S_e	ALL	[2] 0.010			
Low Heat Content	BTU/LB	h	ALL	[2] 8000.			
Particle Density	LB/FT^3	ρ_p	ALL	[2] 32.0			

Notes:

[1]

Class	Description
1	Dead, fine
2	Dead, medium
3	Dead, large
4	Living

[2] Applies to all fuel types.

FUEL CATEGORIES (i)

All fuel types are described by two fuel categories

$i = 1$, dead fuel (1)

$i = 2$, live fuel (2)

FUEL SIZE CLASSES (j)

All fuel types are described by a number of fuel size classes $j=1,N$, where $1 \leq j \leq 6$.

Short Grass and Long Grass

$j = 1$, leaves (3)

$j = 2$, stems (4)

Brush (non-chaparral)

$j = 1$, leaves (5)

$j = 2$, stems < 0.25 inch diameter (6)

$j = 3$, $0.25 \leq$ stems < 0.5 inch diameter (7)

Chamise and Mixed Chaparral

$j = 1$, leaves (8)

$j = 2$, stems < 0.25 inch diameter (9)

$j = 3$, $0.25 \leq$ stems < 0.5 inch diameter (10)

$j = 4$, $0.5 \leq$ stems < 1.0 inch diameter (11)

$j = 5$, $1.0 \leq$ stems < 3.0 inches diameter (12)

FRACTION DEAD (f_D - (weight dead) / (total dry weight))

Short Grass and Long Grass

$$f_D = 0.44 \log_{10}(N_{day}) \qquad (13)$$

N_{day} = number of days since 1 May (used for annual fuels or effects) (14)

Brush, Chamise and Mixed Chaparral

$$f_D = 0.0694 \exp(0.0402A_f) \tag{15}$$

A_f = fuel age in years

TOTAL FUEL LOAD (F_L - grams/meter^2)

For all fuel types F_L is obtained from model FULOD.

FUEL LOADING PER CATEGORY AND SIZE CLASS ($W_o(i,j)$ - grams/meter^2)

Short Grass and Long Grass

Dead

$$W_o(1,1) = 1.3322\ f_D F_L \tag{16}$$

$$W_o(1,2) = 0.0\ f_D F_L \tag{17}$$

Live

$$W_o(2,1) = (0.30 - 0.30\ f_D)F_L \tag{18}$$

$$W_o(2,2) = (0.70 - 0.695\ f_D)F_L \tag{19}$$

Brush

Dead

$$W_o(1,1) = 0.8813\ f_D F_L \tag{20}$$

$$W_o(1,2) = 0.4407\ f_D F_L \tag{21}$$

Live

$$W_o(2,1) = (0.180 - 0.281\ f_D)F_L \tag{22}$$

$$W_o(2,2) = (0.253 - 0.268\ f_D)F_L \tag{23}$$

Chamise and Mixed Chaparral

Dead

$$W_o(1,1) = 0.5977\ f_D F_1 \tag{24}$$

$$W_o(1,2) = 0.9711\ f_D F_L \tag{25}$$

$$W_o(1,3) = 1.9429\ f_D F_L \tag{26}$$

$$W_o(1,4) = 1.9801\ f_D F_L \tag{27}$$

Live

$$W_o(2,1) = (0.1889 - 0.294\ f_D)F_L \tag{28}$$

$$W_o(2,2) = (0.2727 - 0.289\ f_D)F_L \tag{29}$$

FUEL PARTICLE LOW HEAT CONTENT PER CATEGORY AND SIZE CLASS ($h_c(i,j)$ - BTU/lb)

Short Grass and Long Grass

For i = 1,2 and j = 1,2

$$h_c(i,j) = 8000 \tag{30}$$

Brush, Chamise and Mixed Chaparral

Dead

$$h_c(1,1) = 9180 \tag{31}$$

$$h_c(1,j) = 8000,\ j > 1 \tag{32}$$

Live

$$h_c(2,1;2,2) = 9613 - N_{day} + 0.1369N_{day}^2 - 0.000365N_{day}^3 \tag{33}$$

For j > 2

$$h_c(2,j) = 9509 - 10.74N_{day} + 0.1359N_{day}^2 - 0.000405N_{day}^3 \tag{34}$$

FUEL DEPTH (d-meters)

Short Grass

$$d - 0.1451\ \log_{10}(N_{day}) \tag{35}$$

Long Grass

$$d = 0.3631\ \log_{10}(N_{day}) \tag{36}$$

Brush, chamise and chaparral depths are calculated from the total fuel load and depth at 50 years age, and a general growth function.

Brush

$$F_D(50) = 6.0 \tag{37}$$

Chamise

$$F_D(50) = 16.2 \tag{38}$$

Mixed Chaparral

$$F_D(50) = 37.4 \tag{39}$$

Brush, Chamise and Mixed Chaparral

$$d(50) = 5.24 + 0.157F_D(50) - 0.000957F_D^2(50) \tag{40}$$

$$d(A_F) = (0.0492\ A_F - 0.000583\ A_F^2)d(50) \tag{41}$$

FUEL PARTICLE SURFACE-AREA-TO-VOLUME RATIO PER CATEGORY AND SIZE CLASS ($\sigma(i,j)$ - meters^{-1})

Short Grass and Long Grass

For i = 1,2

$$\sigma(i,1) = 1.1482E4 \tag{42}$$

$$\sigma(i,2) = 3.2808E3 \tag{43}$$

Brush

For i = 1,2

$$\sigma(i,1) = 7.2178E3 \tag{44}$$

$$\sigma(i,2) = 2.0997E3 \tag{45}$$

Chamise and Mixed Chaparral

For i = 1,2

$$\sigma(i,1) = 7.2178E3 \tag{46}$$

$$\sigma(i,2) = 2.0997E3 \tag{47}$$

FUEL PARTICLE TOTAL MINERAL CONTENT PER CATEGORY AND SIZE CLASS

($S_T(i,j)$ - mineral weight/fuel weight)

Short Grass and Long Grass

For $i = 1,2$

$$S_T(i,1) = 0.05 \tag{48}$$

$$S_T(i,2) = 0.02 \tag{49}$$

Brush

For $i = 1,2$

$$S_T(i,1) = 0.05 \tag{50}$$

$$S_T(i,2) = 0.02 \tag{51}$$

Chamise and Mixed Chaparral

For $i = 1,2$

$$S_T(i,1) = 0.055 \tag{52}$$

$$S_T(i,j) = 0.025, \text{ for } j > 1 \tag{53}$$

FUEL PARTICLE SILICA-FREE MINERAL CONTENT PER CATEGORY AND SIZE CLASS ($S_E(i,j)$ - Weight of silica-free mineral/fuel weight)

Short Grass and Long Grass

For $i = 1,2$

$$S_E(i,1) = 0.02 \tag{54}$$

$$S_E(i,2) = 0.01 \tag{55}$$

Brush

For $i = 1,2$

$$S_E(i,1) = 0.035 \tag{56}$$

$$S_E(i,2) = 0.015 \tag{57}$$

Chamise and Mixed Chaparral

For $i = 1,2$

$$S_E(i,1) = 0.035 \tag{58}$$

For $j > 1$

$$S_E(i,j) = 0.015 \tag{59}$$

Figure 27. Effect of Relative Humidity-Temperature, Day of Year and Wind Speed Upon Rate of Spread

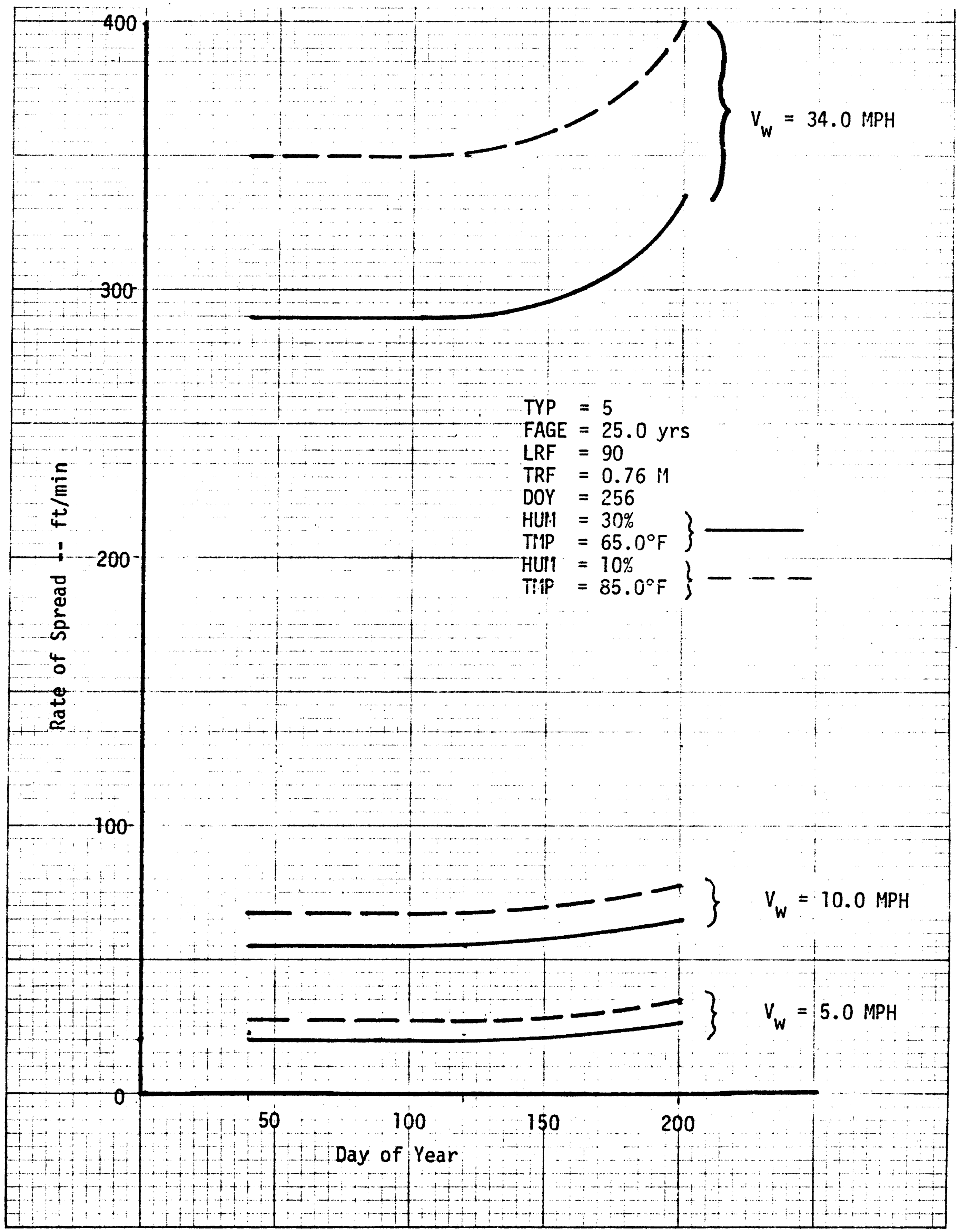
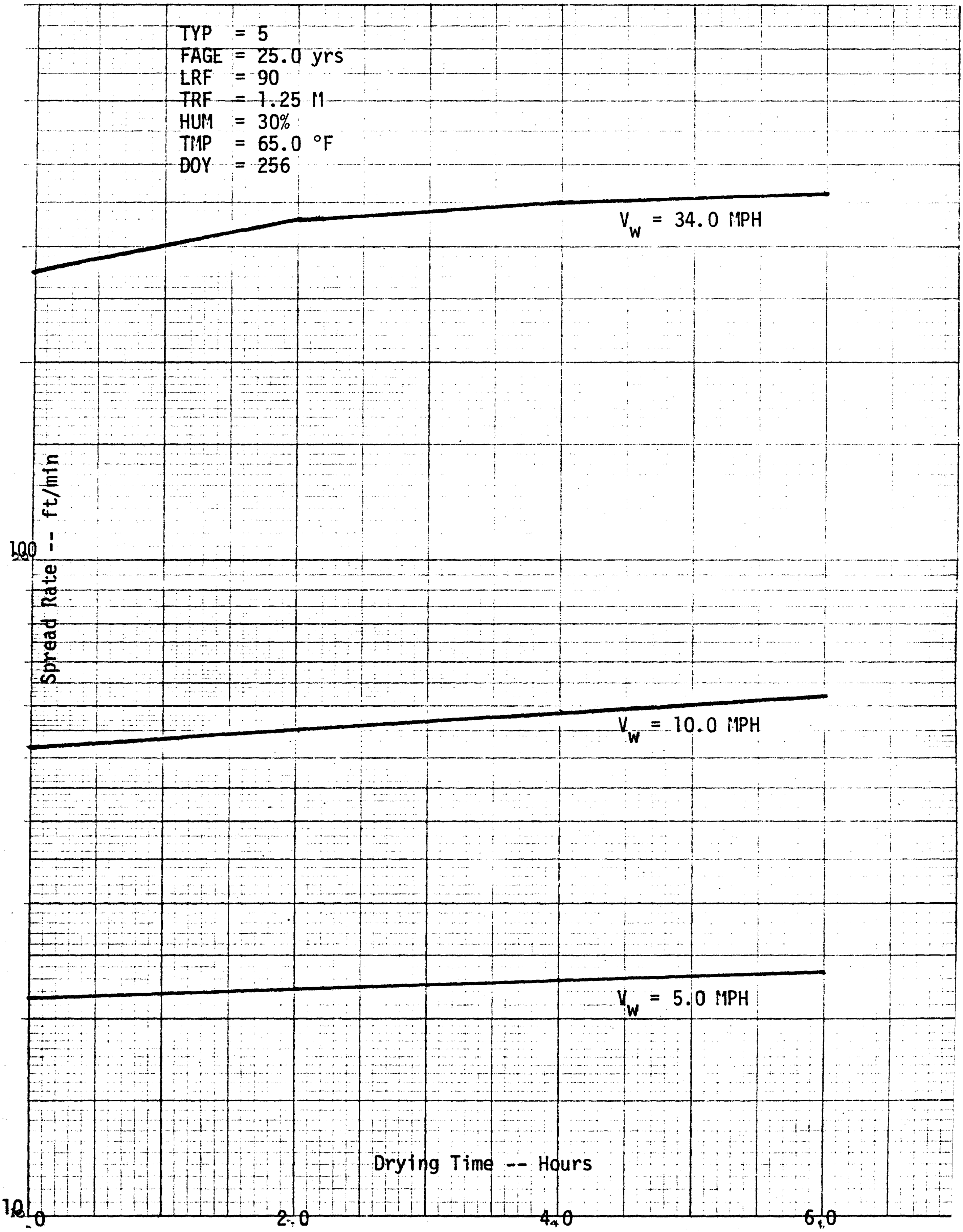


Figure 28. Effect of Dead Fuel Drying Upon Spread Rate



FUEL PARTICLE DENSITY PER CATEGORY AND SIZE CLASS

($\rho_p(i,j)$ - kilogram/liter)

Short Grass and Long Grass

For $i = 1,2$ and $j = 1,2$

$$\rho_p\,(i,j) = 51264 \tag{60}$$

Brush, Chamise and Mixed Chaparral

For $i = 1,2$

$$\rho_p(i,1) = 51264 \tag{61}$$

For $j > 1$

$$\rho_p(i,j) = 76392 \tag{62}$$

4. MODEL RESULTS

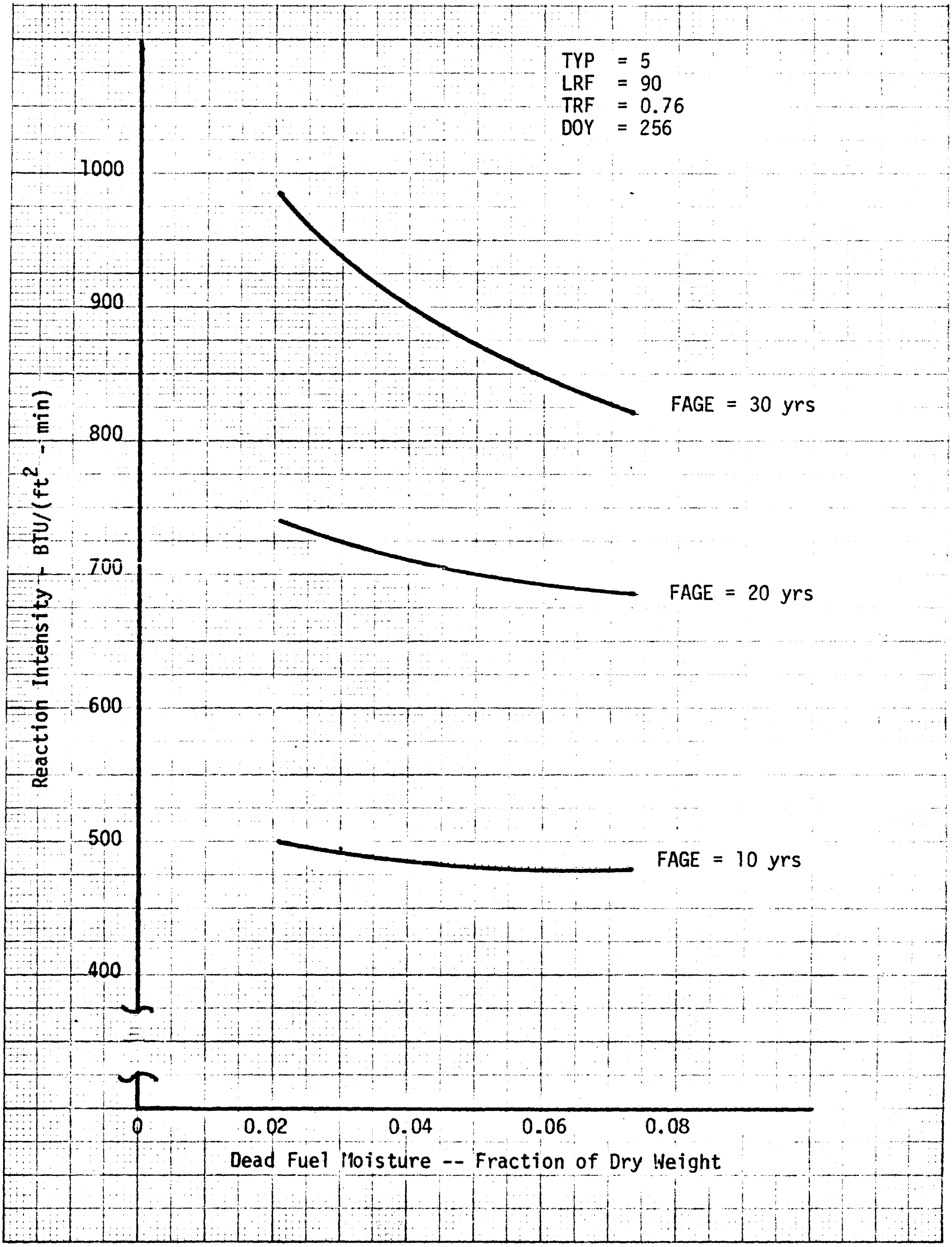
A set of representative model results is presented in Figures 27 through 31 for mixed chaparral. These results are presented to show the way in which selected variables affect rate of spread and reaction intensity.

Figure 27 shows rate of spread (ROS) plotted against day of year with wind speed as a parameter, for two values of relative humidity and temperature. It can be seen that ROS is a weak function of day of year and relative humidity/temperature provided wind speed is less than about 20 mi./hr. At a wind speed of 34 mi./hr. ROS is a strong function of day of year as well as relative humidity/temperature.

Figure 28 shows ROS plotted against drying time for 3 values of wind speed, and other conditions as shown. Drying of fine fuels accounts for the monotonic increase in ROS with drying time.

Figure 29 shows reaction intensity RI plotted against dead fuel moisture fraction with fuel age as a parameter. The effect of dead fuel moisture fraction upon RI is a sensitive function of fuel age because the percent of the plant which constitutes dead fuel is a sensitive function of fuel age particularly for chaparral.

Figure 29. Effects of Dead Fuel Moisture Fraction and Fuel Age Upon Reaction Intensity



Figures 30 and 31 present the reaction intensities and rates of spread for all 5 fuel types, but for a very restricted set of variables. The constant variables for Figures 30 and 31 are:

day of last rainfall	-- 90
total rainfall	-- 0.76 m
fuel age; grass	-- 1 year
other	-- 25 years
relative humidity	-- 0.05 percent
temperature	-- 37.8°C
day of year	-- 256

Figure 30 shows RI plotted against dead fuel moisture fraction for the 5 fuel types currently within the capability of the model. Figure 31 shows ROS plotted against wind speed at mid-fuel height for the 5 fuel classes. The results shown in Figures 30 and 31 agree reasonably with the results of Reference 4.

5. Flow Chart

See Figure 32.

Figure 30. Reaction Intensity for Several Fuel Types as a Function of Dead Fuel Moisture

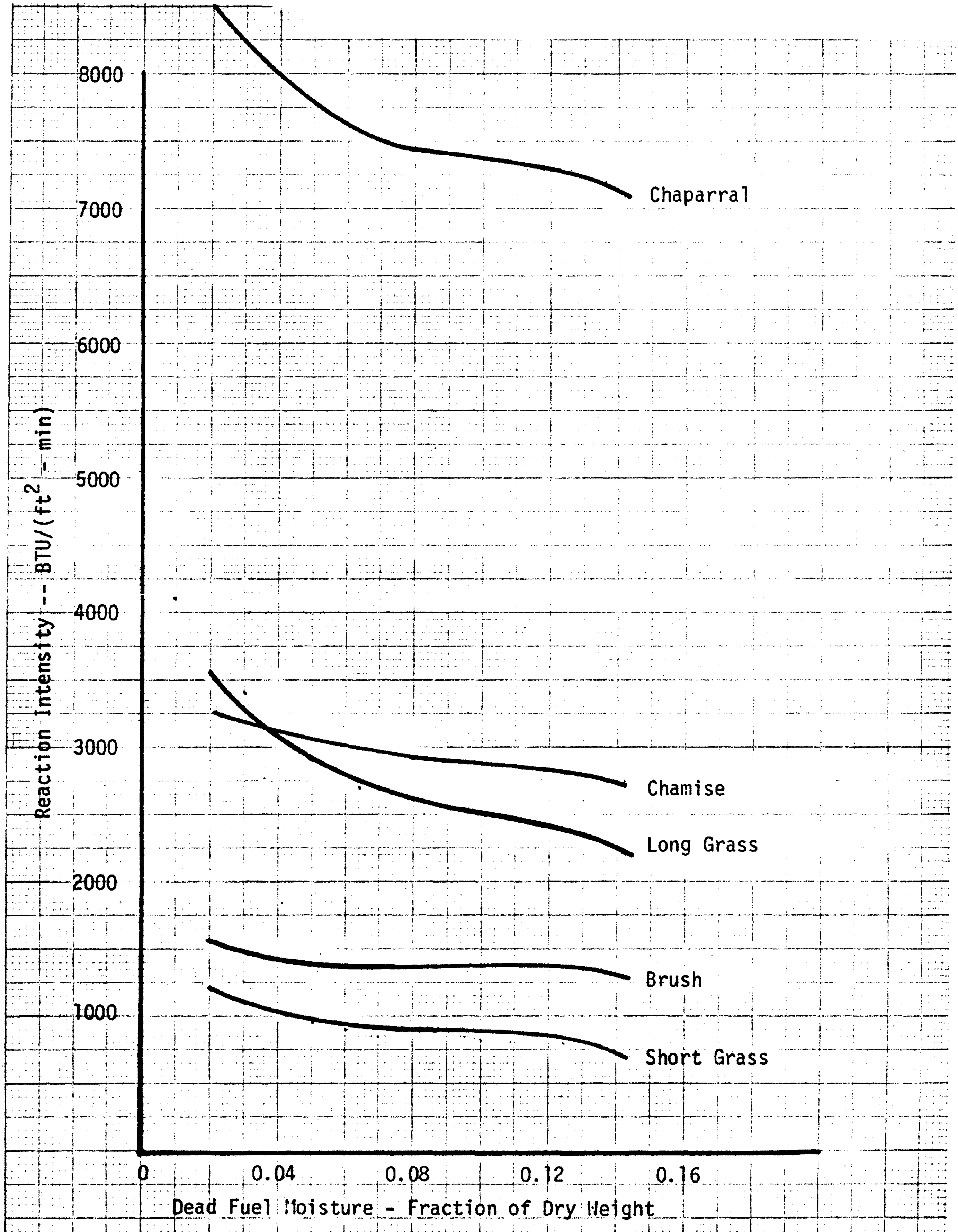


Figure 31. Rate of Spread for Several Fuel Types as a Function of Effective Wind Speed.

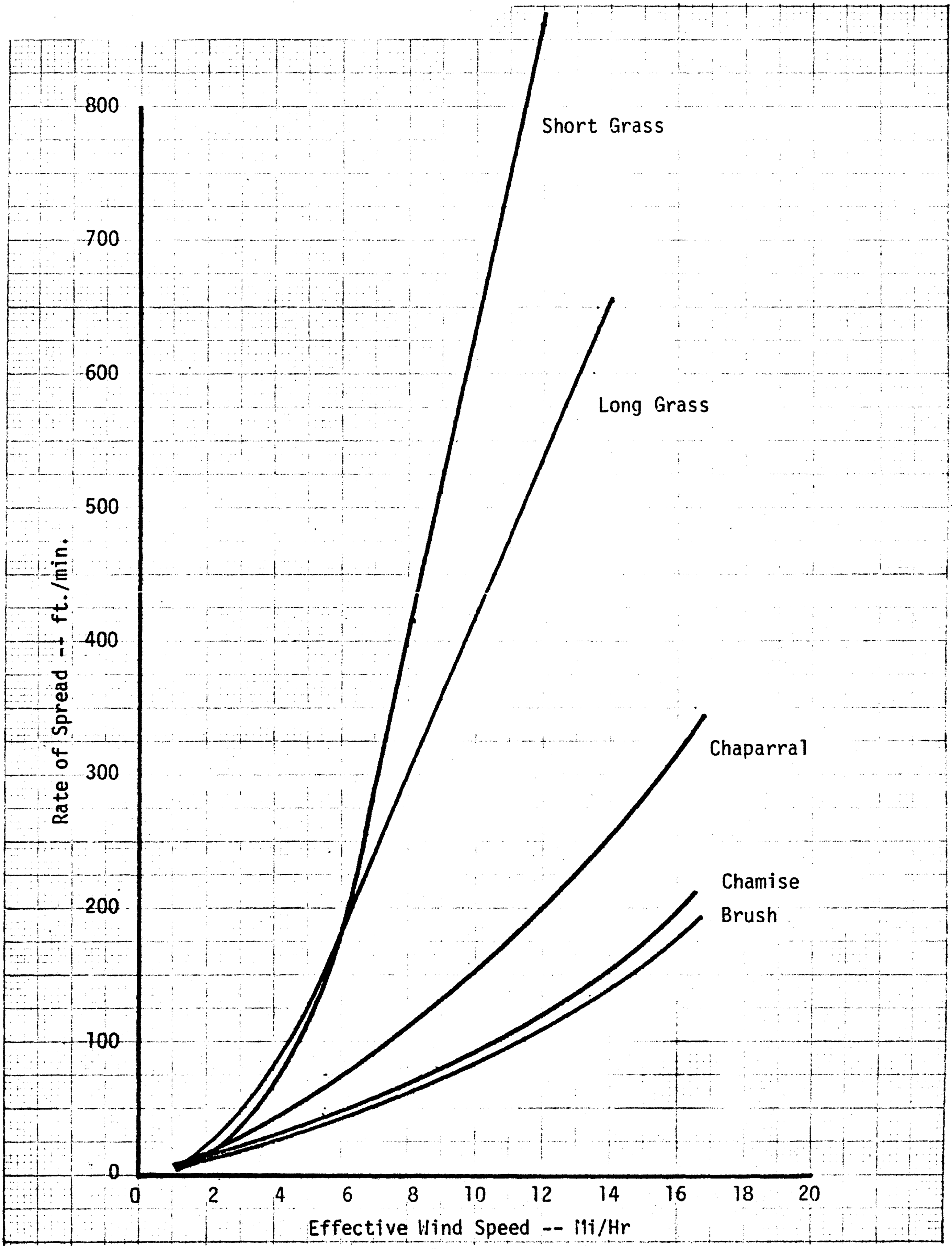
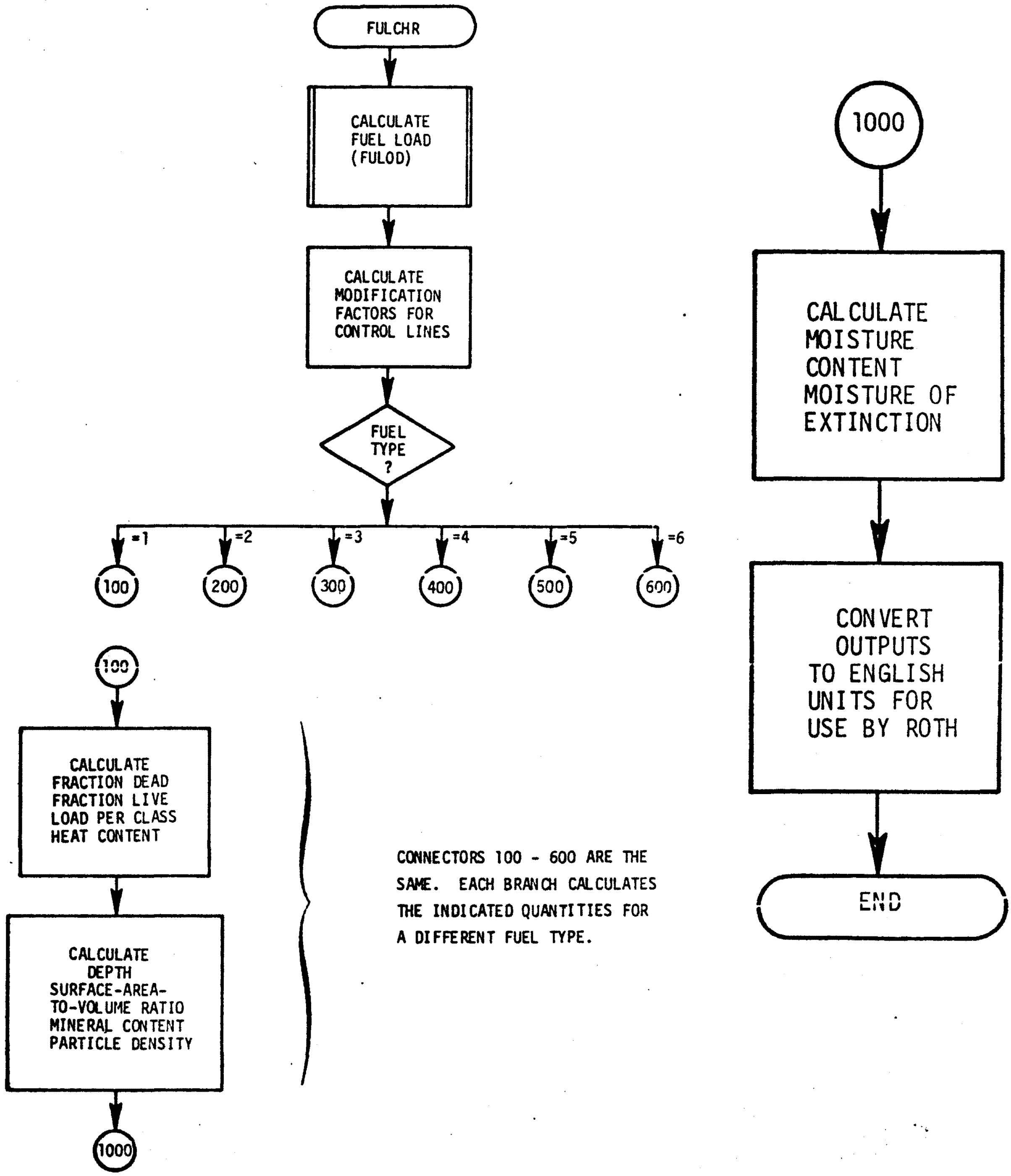


Figure 32. Fuel Characteristics



```
      SUBROUTINE FULCHR(IYR,JDAY,TDOC,TEMP,HREL,WINDSM,IFTYP,AGEI)
C
C     PURPOSE
C            THIS PROGRAM PROVIDES FUEL CHARACTERISTICS
C            FOR USE BY THE RATE-OF-SPREAD MODEL.
C
C     VERSION 1.00      1/21/76      SANDERLIN
C
C            INPUTS FROM LIST
C     IYR     - YEAR OF FIRE (YEARS-1900)
C     JDAY    - DAY OF FIRE (DAY OF FIRE (DAYS SINCE DEC. 31)
C     TDOC    - TIME OF DAY OF CALCULATION (MINUTES FROM MIDNIGHT)
C     TEMP    - TEMPERATURE (DEG. C)
C     HREL    - RELATIVE HUMIDITY (FRACTION)
C     WINDSM  - WIND SPEED (METERS/MIN)
C     IFTYP   - FUEL TYPE
C                =1 SHORT GRASS
C                =2 LONG GRASS
C                =3 BRUSH (NON-CHAPARRAL)
C                =4 CHAMISE(PURE STAND)
C                =5 MIXED CHAPARRAL
C     AGEI    - FUEL AGE
C                FOR IFTYP =1,2 NOT USED (ASSUMED .LT. 1 YR)
C                    IFTYP =3-5 YEARS
C            INPUTS FROM COMMON /SPLMET/
C     JDLRF   - DATE OF LAST RAINFALL
C            OUTPUTS TO COMMON /HINEW/
C     WO      - FUEL LOADING(LBS/FT**2)
C     SIG     - FUEL PARTICLE SURFACE-AREA-TO-VOLUME RATIO(1/FT)
C     ST      - FUEL PARTICLE TOTAL MINERAL CONTENT(LBS
C                MINERALS/LBS OVENDRY WOOD)
C     SE      - FUEL PARTICLE SILICA-FREE MINERAL CONTENT
C                ((LBS MINERALS-LBS SILICA)/LBS OVENDRY WOOD)
C     H       - FUEL PARTICLE LOW HEAT CONTENT(B.T.U./LBS)
C     EMF     - FUEL MOISTURE(LBS MOISTURE/LBS OVENDRY WOOD)
C     RHOP    - FUEL PARTICLE DENSITY(LBS/FT**3)
C     EMX     - FUEL MOISTURE CONTENT OF EXTINCTION(LBS MOISTURE/
C                LBS OVENDRY WOOD)
C     DEPTH   - FUEL DEPTH(FT)
C     WINDS   - WIND SPEED(FT/MIN)
C     M       - NUMBER OF FUEL CATAGORIES(=2)
```

```
C                 CATAGORY 1= DEAD
C                          2= LIVE
C     N       - AN ARRAY OF THE NUMBER OF CLASSES PER CATAGORY(LENGTH=2)
C                        = 1, LEAVES
C                        = 2, STEMS .LT. 0.25 INCH DIA.
C                        = 3, STEMS .GE. 0.25, .LT. 0.5
C                       INCH DIAMETER
C                        = 4, STEMS .GE. 0.5, .LT. 1.0
C                       INCH DIAMETER
C                        = 5, STEMS .GE. 1.0, .LT.  3.0
C                             INCH DIAMETER
C
C     NOTE....... COMMON /BINEW/ IS FOR INPUT TO ROTH ONLY AND IS IN
C             ENGLISH UNITS.
C
C          ROUTINES USED
C     FULOD,FMOISL,FMOISD
C
      COMMON/SPLFET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
      COMMON/BINEW/W0(2,6),SIG(2,6),ST(2,6),SE(2,6),
     & H(2,6),EMF(2,6),RHOP(2,6),EMX(2,6),
     & DEPTH,SLOPE,WINDS,M,N(2)
C
      EQUIVALENCE (N(1),ND01),(N(2),ND02)
C
C
C          NDAY - DAYS SINCE MAY 1 (SINCE APR 30 ON LEAP YEAR)
      NDAY=JDAY-121
      IF(NDAY.LE.0.AND.JDLRF.LE.JDAY)NDAY=JDAY-JDLRF
      IF(JDLRF.GT.JDAY)NDAY=365-JDLRF+JDAY
      DAY = FLOAT(NDAY)
      AGE=ABS(AGEI)
      IF(AGE .GT. 50.0)AGE=50.0
C
C     GET TOTAL FUEL LOAD FROM  FULOD
C
      CALL FULOD(IFTYP,AGE,DAY,FLOD)
C
C     CALCULATE FUEL MODIFICATION FOR CONTROL LINES
C
      F4=AMAX1(0.0,(AMIN1(10.0,0.69+0.61*(FLOD*4.460896-0.75))))
      ETARDC=0.1961292176+F4*(0.0094665707+F4*(-0.0061142318+
     & F4*0.0004030283))
      SEMOD=0.0
      IF(AGEI.LT.0.0)SEMOD=(0.22-ETARDC)
C          MOISTURE MODIFICATION FACTOR
      EMFFAC=1.0
      IF(AGEI.LT.0.0)EMFFAC=
     & AMAX1(1.0,AMIN1(2.0,FLOD*0.2230447804+1.0))
C
C     BRANCH ON FUEL TYPE INDEX
C
      GO TO (100,200,300,400,500),IFTYP
C
C     FUEL TYPE IS SHORT GRASS
C          THERE ARE TWO FUEL CATEGORIES (M)
C                M = 1, DEAD
C                M = 2, LIVE
C
  100 M=2
C          THERE ARE TWO SIZE CLASSES IN
C                EACH FUEL CATEGORY
      N(1)=2
      N(2)=2
C          FRACTION DEAD (GRASS IS CONSIDERED AN ANNUAL)
      FD=0.44*ALOG10(DAY)
```

```
      IF(FD .GT. 1.0)FD=1.0
      IF(AGE .GE. 1.0)FD=1.0
C          FUEL LOAD PER SIZE CLASS PER CATEGORY
C          (KG/METER**2)
C                DEAD FUEL
      WO(1,1)=1.3322*FD*FLOD
      WO(1,2)=0.0*FD*FLOD
C                LIVE FUEL
      WO(2,1)=(0.30-0.2999*FD)*FLOD
      WO(2,2)=(0.70-0.695*FD)*FLOD*0.0
C          HEAT CONTENT (JOULES/KG)
C                DEAD FUEL
      H(1,1)=1.876E7
      H(1,2)=H(1,1)
C                LIVE FUEL
      H(2,1)=9613.0-DAY+0.137*DAY**2+3.65E-4*DAY**3
C          CONVERT FROM BTU/LB TO JOULE/KG
      H(2,1)=2.325E3*H(2,1)
      H(2,2)=H(2,1)
C          FUEL DEPTH (METER)
      DEPTH=0.1451*ALOG10(DAY)
C          FUEL PARTICLE SURFACE-AREA-
C          TO-VOLUME RATIO (1/METER)
C                DEAD FUEL
      SIG(1,1)=1.1482E4
      SIG(1,2)=3.2808E3
C                LIVE FUEL
      SIG(2,1)=1.1482E4
      SIG(2,2)=3.2808E3
C          TOTAL MINERAL CONTENT (UNITS)
C                DEAD FUEL
      ST(1,1)=0.05
      ST(1,2)=0.02
C                LIVE FUEL
      ST(2,1)=0.05
      ST(2,2)=0.02
C          SILICA-FREE MINERAL CONTENT (UNITS)
C                DEAD FUEL
      SE(1,1)=0.01
      SE(1,2)=0.01
C                LIVE FUEL
      SE(2,1)=0.01
      SE(2,2)=0.01
C          FUEL PARTICLE DENSITY (KG/M**3)
C                DEAD FUEL
      RHOP(1,1)=1E3*0.51264
      RHOP(1,2)=1E3*0.51264
C                LIVE FUEL
      RHOP(2,1)=1E3*0.51264
      RHOP(2,2)=1E3*0.51264
      GO TO 1000
C
C     FUEL TYPE IS LONG GRASS
C          THERE ARE TWO FUEL CATEGORIES (M)
C                M=1, DEAD FUEL
C                M=2, LIVE FUEL
  200 M=2
C          THERE ARE TWO SIZE CLASSES IN EACH FUEL
C          CATEGORY, N(M)
C                N=1, LEAVES
C                N=2, STEMS.LT.0.25 INCH DIA.
      N(1)=2
      N(2)=2
C          FRACTION DEAD (GRASS IS CONSIDERED TO BE AN
C          ANNUAL) (UNITS)
      FD=0.44*ALOG10(DAY)
```

```
      IF(FD .GT. 1.0)FD=1.0
      IF(AGE .GE. 1.0)FD=0.9999
C          FUEL LOADING (KG/M**2)
C                 DEAD FUEL
      WO(1,1)=1.7916*FD*FLOD
      WO(1,2)=0.0*FD*FLOD
C                 LIVE FUEL
      WO(2,1)=(0.30-0.2999*FD)*FLOD
      WO(2,2)=(0.70-0.695*FD)*FLOD*0.0
C          HEAT CONTENT
C                 DEAD FUEL
      H(1,1)=8000.*2.3254E3
      H(1,2)=H(1,1)
C                 LIVE FUEL
      H(2,1)=2.3254E3*(9613.-DAY+0.1369*DAY**2+3.65E-4*
     &DAY**3)
      H(2,2)=H(2,1)
C          FUEL DEPTH
      DEPTH=0.3631*ALOG10(DAY)
C          FUEL PARTICLE SURFACE-AREA-TO VOLUME
C          RATIO (1/METERS)
C                 DEAD FUEL
      SIG(1,1)=4.921E3
      SIG(1,2)=2.625E3
C                 LIVE FUEL
      SIG(2,1)=4.921E3
      SIG(2,2)=2.625E3
C          TOTAL MINERAL CONTENT (UNITS)
C                 DEAD FUEL
      ST(1,1)=0.05
      ST(1,2)=0.02
C                 LIVE FUEL
      ST(2,1)=0.05
      ST(2,2)=0.02
C          SILICA-FREE MINERAL CONTENT (UNITS)
C                 DEAD FUEL
      SE(1,1)=0.01
      SE(1,2)=0.01
C                 LIVE FUEL
      SE(2,1)=0.01
      SE(2,2)=0.01
C          FUEL PARTICLE DENSITY (KG/LITER)
C                 DEAD FUEL
      RHOP(1,1)=0.51264*1E3
      RHOP(1,2)=0.51264*1E3
C                 LIVE FUEL
      RHOP(2,1)=0.51264*1E3
      RHOP(2,2)=0.51264*1E3
C
      GO TO 1000
C
C     FUEL TYPE IS BRUSH (NON-CHAPARRAL)
C          THERE ARE TWO FUEL CATEGORIES
C                 M =1, DEAD
C                 M = 2, LIVE
  300 M=2
C          THERE ARE TWO SIZE CLASSES OF DEAD FUELS
C          ,N(1), AND THREE SIZE CLASSES OF LIVE FUELS,
C          N(2). LEAVES ARE OMITTED FROM DEAD FUEL
C                 N=1, LEAVES
C                 N=2, STEMS.LT.0.25 INCH DIA.
C                 N=3, STEMS.GE.0.25,.LT.0.50 INCH DIA.
      N(1)=3
      N(2)=3
C          FRACTION DEAD (UNITS)
      FD=0.0694*EXP(0.0402*AGE)
```

```
C            FUEL LOADING (KG/METER**2)
C                      DEAD FUEL
      WO(1,1)=0.8813*FD*FLOD
      WO(1,2)=0.4407*FD*FLOD
      WO(1,3)=0.0*FD*FLOD
C                      LIVE FUEL
      WO(2,1)=(0.196-0.305*FD)*FLOD*0.920
      WO(2,2)=(0.242-0.256*FD)*FLOD*1.0488
      WO(2,3)=(0.561-0.420*FD)*FLOD*0.0
C            HEAT CONTENT
C                      DEAD FUEL
      H(1,1)=2.3254E3*9140.
      H(1,2)=2.3254E3*8000.
      H(1,3)=H(1,2)
C                      LIVE FUEL
      H(2,1)=2.3254E3*(9613.-DAY+0.1369*DAY**2+3.65E-4*
     &DAY**3)
      H(2,2)=H(2,1)
      H(2,3)=2.3254E3*(9509.-10.74*DAY+0.1359*DAY**2
     &-4.05E-4*DAY**3)
C            FUEL DEPTH-CALCULATED FROM FUEL LOAD AT
C            FIFTY YEARS AGE AND, A GENERAL GROWTH
C            FUNCTION
C
C                      FUEL LOAD AT 50 YEARS AGE
      W50=6.0
C                      FUEL DEPTH AT 50 YEARS AGE
      D50=(5.24+0.157*W50-9.57E-4*W50**2)*0.1068
C            FUEL DEPTH AT AGE (METERS)
      DEPTH=D50*(4.92E-2*AGE-4.82E-4*AGE**2)
C            FUEL SURFACE-AREA-TO-VOLUME RATIO (1/METER)
C                      DEAD FUEL
      SIG(1,1)=6.10E3
      SIG(1,2)=3.32E1
      SIG(1,3)=3.5761E2
C                      LIVE FUEL
      SIG(2,1)=6.5617E3
      SIG(2,2)=2.0997E3
      SIG(2,3)=3.5761E3
C            TOTAL MINERAL CONTENT (UNITS)
C                      DEAD FUEL
      ST(1,1)=0.05
      ST(1,2)=0.02
      ST(1,3)=0.02
C                      LIVE FUEL
      ST(2,1)=0.05
      ST(2,2)=0.02
      ST(2,3)=0.02
C            SILICA-FREE MINERAL CONTENT (UNITS)
C                      DEAD FUEL
      SE(1,1)=0.035
      SE(1,2)=0.015
      SE(1,3)=0.015
C                      LIVE FUEL
      SE(2,1)=0.035
      SE(2,2)=0.015
      SE(2,3)=0.015
C            FUEL PARTICLE DENSITY (KG/M**3)
C                      DEAD FUEL
      RHOP(1,1)=1E3*0.51264
      RHOP(1,2)=1E3*0.73692
      RHOP(1,3)=1E3*0.73692
C                      LIVE FUEL
      RHOP(2,1)=1E3*0.51264
      RHOP(2,2)=1E3*0.73692
      RHOP(2,3)=1E3*0.73692
```

```
C
      GO TO 1000
C
C     FUEL TYPE IS CHAMISE (PURE STAND)
C          THERE ARE TWO FUEL CATEGORIES (M)
C                 M=1, DEAD
C                 M=2, LIVE
  400 M=2
C          THERE ARE FOUR SIZE CLASSES,N(1),FOR
C          DEAD FUELS, AND FIVE SIZE CLASSES,N(2), FOR
C          LIVE FUELS. LEAVES ARE OMITTED FROM
C          DEAD FUELS.
C                 N=1, LEAVES
C                 N=2, STEMS.LT.0.25 INCH DIA.
C                 N=3, STEMS.GE.0.25, LT.0.50 INCH DIA.
C                 N=4, STEMS.GE.0.50,.LT.1.0 INCH DIA.
C                 N=5, STEMS.GE.1.0, .LT.3.0 INCH DIA.
      N(1)=5
      N(2)=5
C          FRACTION DEAD (UNITS)
      FD=0.0694*EXP(0.0402*AGE)
C          FUEL LOAD (KG/METER**2)
C                 DEAD FUEL
      WO(1,1)=0.5977*FD*FLOD
      WO(1,2)=0.9711*FD*FLOD
      WO(1,3)=1.9429*FD*FLOD
      WO(1,4)=1.9601*FD*FLOD
      WO(1,5)=0.0*FD*FLOD
C                 LIVE FUEL
      WO(2,1)=(0.1957-0.305*FD)*FLOD*0.9651
      WO(2,2)=(0.2416-0.256*FD)*FLOD*1.1288
      WO(2,3)=(0.1918-0.256*FD)*FLOD*0.0
      WO(2,4)=(0.2648-0.050*FD)*FLOD*0.0
      WO(2,5)=(0.2036-0.114*FD)*FLOD*0.0
C          HEAT CONTENT
C                 DEAD FUEL
      H(1,1)=2.3254E3*9160.
      H(1,2)=2.3254E3*8000.
      H(1,3)=2.3254E3*8000.
      H(1,4)=2.3254E3*8000.
      H(1,5)=2.3254E3*8000.
C                 LIVE FUEL
      H(2,1)=2.3254E3*(9613.0-DAY+0.1369*DAY**2
     &+3.65E-4*DAY**3)
      H(2,2)=H(2,1)
      H(2,3)=2.3254E3*(9509.-DAY+0.1359*DAY**2-
     &4.05E-4*DAY**3)
      H(2,4)=H(2,3)
      H(2,5)=H(2,3)
C      FUEL DEPTH - CALCULATED FROM FUEL LOAD
C      AT 50YEARS AGE AND A GENERAL GROWTH FUNCTION.
C           FUEL LOAD AT 50 YEARS AGE
      W50=16.2
C           FUEL LOAD AT 50 YEARS AGE
      D50=(5.24+0.157*W50-9.57E-4*W50**2)*0.193
C           FUEL DEPTH AT AGE
      DEPTH=D50*(4.92E-2*AGE-4.82E-4*AGE**2)
C       FUEL SURFACE-AREA-TO-VOLUME RATIO
C           DEAD FUEL
      SIG(1,1)=7.23178E3
      SIG(1,2)=2.0997E3
      SIG(1,3)=4.1667E2
      SIG(1,4)=2.0013E2
      SIG(1,5)=8.6583E1
C           LIVE FUEL
      SIG(2,1)=SIG(1,1)
```

```
      SIG(2,2)=SIG(1,2)
      SIG(2,3)=SIG(1,3)
      SIG(2,4)=SIG(1,4)
      SIG(2,5)=SIG(1,5)
C             TOTAL MINERAL CONTENT (UNITS)
C                     DEAD FUEL
      ST(1,1)=0.055
      ST(1,2)=0.025
      ST(1,3)=0.025
      ST(1,4)=0.025
      ST(1,5)=0.025
C                     LIVE FUEL
      ST(2,1)=0.055
      ST(2,2)=0.025
      ST(2,3)=0.025
      ST(2,4)=0.025
      ST(2,5)=0.025
C             SILICA-FREE MINERAL CONTENT (UNITS)
C                     DEAD FUEL
      SE(1,1)=0.035
              SE(1,2)=0.015
      SE(1,3)=0.015
      SE(1,4)=0.015
      SE(1,5)=0.015
C                     LIVE FUEL
      SE(2,1)=0.035
      SE(2,2)=0.015
      SE(2,3)=0.015
      SE(2,4)=0.015
      SE(2,5)=0.015
C             FUEL PARTICLE DENSITY (KG/M**3)
C                     DEAD FUEL
      RHOP(1,1)=0.51264*1E3
      RHOP(1,2)=0.73692*1E3
      RHOP(1,3)=0.73692*1E3
      RHOP(1,4)=0.73692*1E3
      RHOP(1,5)=0.73692*1E3
C                     LIVE FUEL
      RHOP(2,1)=0.51264*1E3
      RHOP(2,2)=0.73692*1E3
      RHOP(2,3)=0.73692*1E3
      RHOP(2,4)=0.73692*1E3
      RHOP(2,5)=0.73692*1E3
C
      GO TO 1000
C
C     FUEL TYPE IS MIXED CHAPARRAL
C             THERE ARE TWO FUEL CATAGORIES (M)
C                     M=1, DEAD
  500 M=2
C             THERE ARE FOUR SIZE CLASSES FOR DEAD FUEL,
C             AND FIVE SIZE CLASSES FOR LIVE FUEL.
C             LEAVES ARE OMITTED FROM DEAD FUELS
C                     N=1, LEAVES
C                     N=2, STEMS.LT.0.25 INCH DIA.
C                     N=3, STEMS.GE. 0.25, 9LT.0.50 INCH DIA.
C                     N=4, STEMS.GE.0.50,.LT.1.0 INCH DIA.
C                     N=5, STEMS.GT.1.0,.LT.3.0 INCH DIA.
C
      N(1)=5
      N(2)=5
C     FRACTION DEAD (UNITS)
      FD=0.0694*EXP(0.0402*AGE)
C             FUEL LOAD (KG/METER**2)
C                     DEAD FUEL
      WO(1,1)=0.5977*FD*FLOD
```

```
      WO(1,2)=0.9711*FD*FLOD
      WO(1,3)=1.9429*FD*FLOD
      WO(1,4)=1.9801*FD*FLOD
      WO(1,5)=0.0*FD*FLOD
C                 LIVE FUEL
      WO(2,1)=(0.1957-0.305*FD)*FLOD*0.9651
      WO(2,2)=(0.2416-0.256*FD)*FLOD*1.1288
      WO(2,3)=(0.1918-0.256*FD)*FLOD*0.0
      WO(2,4)=(0.2648-0.050*FD)*FLOD*0.0
      WO(2,5)=(0.1036-0.114*FD)*FLOD*0.0
C            HEAT CONTENT (JOULES/KG)
C                 DEAD FUEL
      H(1,1)=2.3254E3*9180.
      H(1,2)=2.3254E3*8000.
      H(1,3)=2.3254E3*3000.
      H(1,4)=2.3254E3*8000.
      H(1,5)=2.3254E3*8000.
C                 LIVE FUEL
      H(2,1)=2.3254E3*(9613.-DAY+0.1369*DAY**2
     &+3.65E-4*DAY**3)
      H(2,2)=H(2,1)
      H(2,3)=2.3254E3*(9509.-10.74*DAY+0.1359*DAY**2
     &-4.05E-4*DAY**3)
      H(2,4)=H(2,3)
      H(2,5)=H(2,3)
C            FUEL DEPTH-CALCULATED FROM FUEL LOAD AT
C            50 YEARS AGE, AND A GENERAL FUEL GROWTH
C            FUNCTION.
C                 FUEL LOAD AT 50 YEARS AGE
      W50=37.4
C                 FUEL DEPTH AT 50 YEARS AGE
      D50=(5.24+0.157*W50-9.57E-4*W50**2)*0.2492
C            FUEL DEPTH AT AGE (UNITS)
      DEPTH=D50*(4.92E-2*AGE-4.82E-4*AGE**2)
C            FUEL PARTICLE SURFACE-AREA-TO-VOLUME
C            RATIO  (1/METERS)
C                 DEAD FUEL
      SIG(1,1)=7.2178E3
      SIG(1,2)=2.0097E3
      SIG(1,3)=4.1667E2
      SIG(1,4)=2.0013E2
      SIG(1,5)=8.8583E1
C                 LIVE FUEL
      SIG(2,1)=7.2178E3
      SIG(2,2)=2.0097E3
      SIG(2,3)=4.1667E2
      SIG(2,4)=2.0013E2
      SIG(2,5)=8.8583E1
C            TOTAL MINERAL CONTENT ( UNITS)
C                 DEAD FUEL
      ST(1,1)=0.055
      ST(1,2)=0.025
      ST(1,3)=0.025
      ST(1,4)=0.025
      ST(1,5)=0.025
C                 LIVE FUEL
      ST(2,1)=0.055
      ST(2,2)=0.025
      ST(2,3)=0.025
      ST(2,4)=0.025
      ST(2,5)=0.025
C            SILICA-FREE MINERAL CONTENT
C                 DEAD FUEL
      SE(1,1)=0.035
      SE(1,2)=0.015
      SE(1,3)=0.015
```

```
      SE(1,4)=0.015
      SE(1,5)=0.015
C                 LIVE FUEL
      SE(2,1)=0.035
      SE(2,2)=0.015
      SE(2,3)=0.015
      SE(2,4)=0.015
      SE(2,5)=0.015
C          FUEL PARTICLE DENSITY (KG/M**3)
C                 DEAD FUEL
      RHOP(1,1)=0.51264*1E3
      RHOP(1,2)=0.73692*1E3
      RHOP(1,3)=0.73692*1E3
      RHOP(1,4)=0.73692*1E3
      RHOP(1,5)=0.73692*1E3
C                 LIVE FUEL
      RHOP(2,1)=0.51264*1E3
      RHOP(2,2)=0.73692*1E3
      RHOP(2,3)=0.73692*1E3
      RHOP(2,4)=0.73692*1E3
      RHOP(2,5)=0.73692*1E3
C
C     GO TO 1000
C
 1000 CONTINUE
C
C      CALCULATE FUEL MOISTURE CONTENT FOR N(2)
C      CLASSES OF LIVE FUELS AND N(1) CLASSES
C      OF DEAD FUELS
C
C           GET MOISTURE FRACTION OF LIVE FUELS
      CALL FMOISL(JDAY,IYR,N,EMF)
C      GET MOISTURE FRACTION OF DEAD FUELS
      CALL FMOISD(TOOC,TEMP,HREL,WINDSM,N,FMF)
      NX=N(1)+1
C
C      FILL UNUSED ARRAY ELEMENTS WITH ZEROES
C
      DO 1020 I=1,M
      DO 1010 J=NX,6
      WO(I,J)=0.0
      SIG(I,J)=0.0
      ST(I,J)=0.0
      SE(I,J)=0.0
      H(I,J)=0.0
      RHOP(I,J)=0.0
      EMF(I,J)=0.0
      FMX(I,J)=0.0
 1010 CONTINUE
 1020 CONTINUE
C      MOISTURE OF EXTINCTION FOR DEAD FUELS
      DO 1030 J=1,ND01
      EMX(1,J)=0.20
 1030 CONTINUE
C           MOISTURE OF EXTINCTION FOR LIVE FUELS
      ALPHA=(WO(2,1)+WO(2,2))/(WO(1,1)+WO(1,2)+WO(2,1)+WO(2,2))
      EMXL=2.9*(1.0-ALPHA)/ALPHA*(1.0-3.3333*EMF(1,1))-2.66
      IF(EMXL .LT. 0.30)EMXL=0.30
      DO 1040 J=1,ND02
      EMX(2,J)=EMXL
 1040 CONTINUE
C
C      CONVERT TO ENGLISH UNITS
C
      DO 1060 I=1,M
      DO 1050 J=1,6
```

```
      WO(I,J)=0.2048*WO(I,J)
      SIG(I,J)=0.305*SIG(I,J)
C          MODIFY SE FOR CONTROL LINES
      SE(I,J)=SE(I,J)+SEMOD
C          MODIFY EMF FOR CONTROL LINES
      EMF(I,J)=EMF(I,J)*EMFFAC
      H(I,J)=4.3E-4*H(I,J)
      RHOP(I,J)=6.243E-2*RHOP(I,J)
 1050 CONTINUE
 1060 CONTINUE
      DEPTH =3.2808*DEPTH
      WINDS=3.2808*WINDSM
 1070 CONTINUE
      RETURN
      END
```

25.0 SPREAD RATE MODEL (ROTH)*

1. Purpose

This routine computes spread rates for wildland fires.

2. Arguments

<u>Input</u>

The inputs for this routine are generated by subroutine FULCHR.

<u>Output</u>

RO	-	no slope, no wind spread rate (meters/minute)
PS	-	slope factor affecting spread rate
PW	-	wind factor affecting spread rate

3. Procedure

The procedure used for this routine was extracted from a mathematical model by R. C. Rothermel, pp. 28-35.

4. Comments

None.

5. Flow Chart

See Figure 33.

---

*This routine, in its original form, was supplied by the Pacific Southwest Fire Laboratory at Riverside, California.

Figure 33. Spread Rate

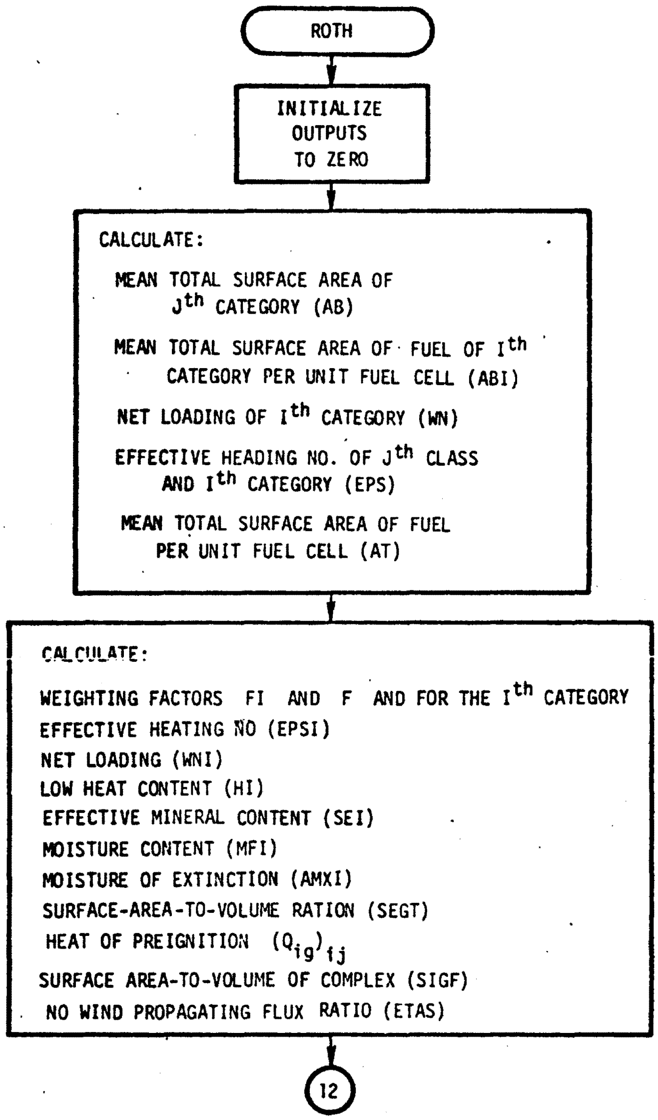
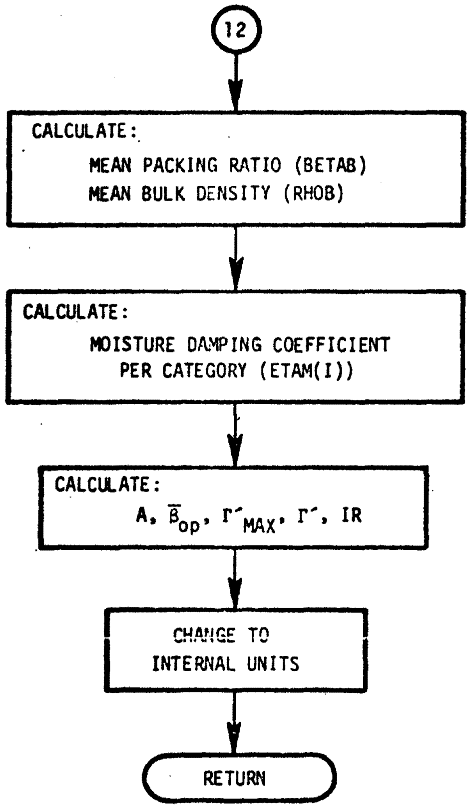


Figure 33. Spread Rate (Cont'd)



```
      SUBROUTINE ROTH(RO,PS,PW)
C
C     ROTH CALCULATES THE RATE OF SPREAD BASED ON ROTHERMAL
C     VERSION 2.00      1/13/76
C
C     PROGRAMMER - VAN GELDER
C     COMMENTS AND SLIGHT MODIFICATIONS  -  JOHN SUNDERSON, JR.
C
      COMMON /BINEW / WO(2,6),SIG(2,6),ST(2,6),SE(2,6),H(2,6),AMF(2,6),
     & RHOP(2,6),AMX(2,6),DEPTH,SLOPE,WIND,M,N(2)
      REAL IR,MEXTL,MFI
      DIMENSION HI(2),AB(2,6),ABI(2),EPS(2,6),EPSI(2),WN(2,6),
     & WNI(2),SEI(2),F(2,6),FI(2),ETAS(2),ETAM(2),QIG(2,6),MFI(2),
     & SIGT(2)
      DIMENSION AMXI(2)
      NCAT=M
C NOTE....NUMBER OF CLASSES ASSUMED SAME FOR BOTH CATAGORIES
      NCLS=N(1)
C
      FML=0.
      FMT=0.
      DO 6 JX=1,NCLS
      IF(SIG(2,JX).LT.192.)GO TO 6
      FML=FML+WO(2,JX)
      DO 5 IX=1,NCAT
      FMT=FMT+WO(IX,JX)
    5 CONTINUE
    6 CONTINUE
      DO 7 IX=1,NCAT
      EPSI(IX)=0.
      ABI(IX)=0.
      WNI(IX)=0.
      HI(IX)=0.
      SEI(IX)=0.
      AMXI(IX)=0.0
      MFI(IX)=0.
    7 SIGT(IX)=0.
      AT=0.
      IR=0.
      SIGF=0.
      BETAB=0.
      RHOB=0.
      EPSW=0.
      RBEPSQ=0.
      RBEPSG=0.
      PW=0.0
      PS=0.0
      RO=0.0
C
C     CALCULATE
C     1) MEAN TOTAL SURFACE AREA OF FUEL OF JTH CLASS AND ITH CATEGORY
```

```
C          PER UNIT FUEL CELL (AB)
C       2) MEAN TOTAL SURFACE AREA OF FUEL OF ITH CATEGORY PER UNIT FUEL
C          CELL (ABI)
C       3) NET LOADING OF ITH CATEGORY (WN)
C       4) EFFECTIVE HEATING NUMBER OF JTH CLASS AND ITH CATEGORY (EPS)
C       5) MEAN TOTAL SURFACE AREA OF FUEL PER UNIT FUEL CELL (AT)
        DO 9 I=1,NCAT
        DO 8 J=1,NCLS
        AB(I,J)=0.
        IF(RHOP(I,J).NE.0.0)AB(I,J)=SIG(I,J)*WO(I,J)/RHOP(I,J)
        ABI(I)=ABI(I)+AB(I,J)
        WN(I,J)=WO(I,J)/(1.0+ST(I,J))
        EPS(I,J)=0.
        IF(SIG(I,J).GT.0.)EPS(I,J)=EXP(-138./SIG(I,J))
      8 CONTINUE
      9 AT=AT+ABI(I)
C
C       PERFORM CHECKS FOR IMPOSSIBLE DATA
C
        IF(DEPTH .LE. 0.0)GO TO 15
        IF(AT .LE. 0.0)GO TO 15
        DO 11 I=1,NCAT
        FI(I)=ABI(I)/AT
        DO 10 J=1,NCLS
        F(I,J)=0.0
        IF(ABI(I).NE.0.0)F(I,J)=AB(I,J)/ABI(I)
        EPSI(I)=EPSI(I)+F(I,J)*EPS(I,J)
        WNI(I)=WNI(I)+F(I,J)*WN(I,J)
        HI(I)=HI(I)+F(I,J)*H(I,J)
        SEI(I)=SEI(I)+F(I,J)*SE(I,J)
        MFI(I)=MFI(I)+F(I,J)*AMF(I,J)
        AMXI(I)=AMXI(I)+F(I,J)*AMX(I,J)
        SIGT(I)=SIGT(I)+F(I,J)*SIG(I,J)
        QIG(I,J)=250.+1116.*AMF(I,J)
        IF(RHOP(I,J).EQ.0.0)GO TO 10
        BETAB=BETAB+WO(I,J)/RHOP(I,J)
     10 RHOB=RHOB+WO(I,J)
        SIGF=SIGF+FI(I)*SIGT(I)
        ETAS(I)=0.
        IF(SEI(I).NE.0.)ETAS(I)=.174*SEI(I)**(-.19)
        IF(ETAS(I).GT.1.)ETAS(I)=1.
     11 CONTINUE
C       AMXI(1)=AMX(1,1)
C       AMXI(2)=AMX(2,1)
        BETAB=BETAB/DEPTH
        RHOB=RHOB/DEPTH
C
C       CALCULATE MOISTURE DAMPING COEFFICIENT OF EACH CATEGORY
C
        DO 110 I=1,NCAT
        RM=1.0
        IF(AMXI(I).NE.0.0)RM=MFI(I)/AMXI(I)
        IF(RM.GT.1.0)RM=1.0
        ETAM(I)=1.+RM*(-2.59+RM*(5.11-3.52*RM))
    110 CONTINUE
C
C
C
C
        AEXP=133.*SIGF**(-.7913)
        BOP=3.348*SIGF**(-.8189)
        GPMAX=1./(495.*SIGF**(-1.5)+.0594)
        BRAT=BETAB/BOP
        GP=GPMAX*(BRAT**AEXP)*EXP(AEXP*(1.-BRAT))
        DO 14 I=1,NCAT
        EPSQ=0.
```

```
C
      IR=IR+FI(I)*WNI(I)*HI(I)*ETAS(I)*ETAM(I)
      DO 13 J=1,NCLS
   13 EPSQ=EPSQ+F(I,J)*EPS(I,J)*QIG(I,J)
   14 RBEPSQ=RBEPSQ+FI(I)*EPSQ
      RBEPSQ=RBEPSQ*RHOB
      IR=IR*GP
      IF(IR.EQ.0.0)GO TO 15
      XI=EXP((.792+.681*SIGF**(.5))*(BETAB+.1))/(192.+.2595*SIGF)
      RO=IR*XI/RBEPSQ
      B=.02526*SIGF**(.54)
      C=7.47*EXP(-.133*SIGF**(.55))
      E=.715*EXP(-.000359*SIGF)
      PWI=C*BRAT**(-E)
      PSI=5.275*BETAB**(-.3)
      U=WIND/2.
      IF(U/IR.GT.0.9)U=.9*IR
      PW=PWI*U**B
      PS=PSI*SLOPE*SLOPE
C         CONVERT RATE TO METERS/MIN
      RO=RO*0.3048
   15 RETURN
      END
```

26.0 LIVE FUEL MOISTURE

The state of knowledge regarding the dynamics of fuel moisture in both live and dead fuels is relatively undeveloped. There are, however, sufficient data and theory to define an intuitive model. Such a model exhibits the correct qualitative response to the parameters of interest, although the quantitative relation to reality may be relatively poor. For these reasons, the model concept is described in considerable detail, but is applied in a much simplified form.

Moisture Content of Live Fuel

The live fuel moisture model is based on observations of chamise. This is the dominant species of chaparral in Southern California, being found in about 70 percent of all the stands in the state.[7]

The moisture content of living chamise depends largely on the physiological activity of the plant. During the late fall and winter months when the shrubs are dormant, the moisture content remains relatively constant, typically between 80 and 110 percent of the dry weight. As the spring growing season approaches, the moisture content of the mature foliage and fine material rises slowly. The new growth, usually reaching its maximum development between March and May, has a very high moisture content - sometimes over 200 percent of the dry weight. The moisture content of both the mature and new growth declines during the summer and reaches a minimum in September or October. A typical example of moisture content for new and mature growth foliage is shown in Figure 34.[8] A description of live fuel moisture content for chamise which has been used in a dynamic fuel model is shown in Figure 35.[5] The moisture content of chaparral in May (at the end of the rainy season) is dependent to some extent upon the amount of rainfall that occurred during the winter months. Rainfall for an average year and a dry year are given in Reference 7.

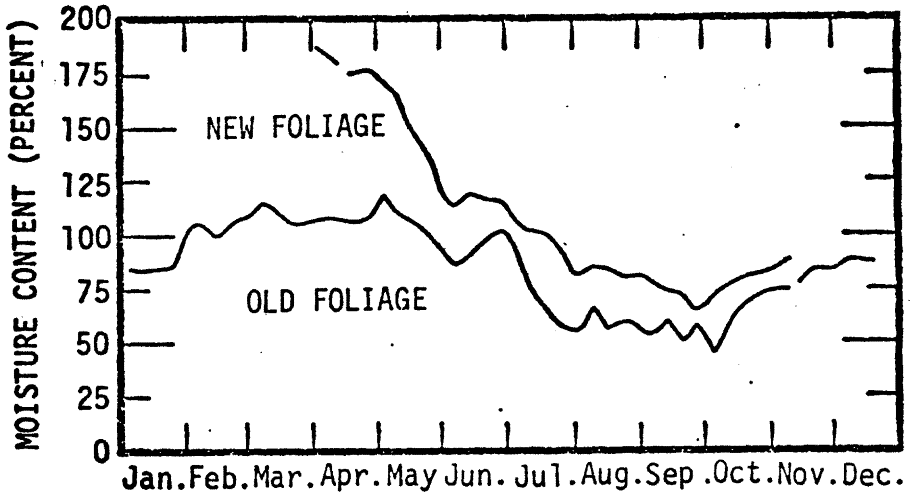


FIGURE 34 Typical Annual Moisture Content of Chamise Foliage.

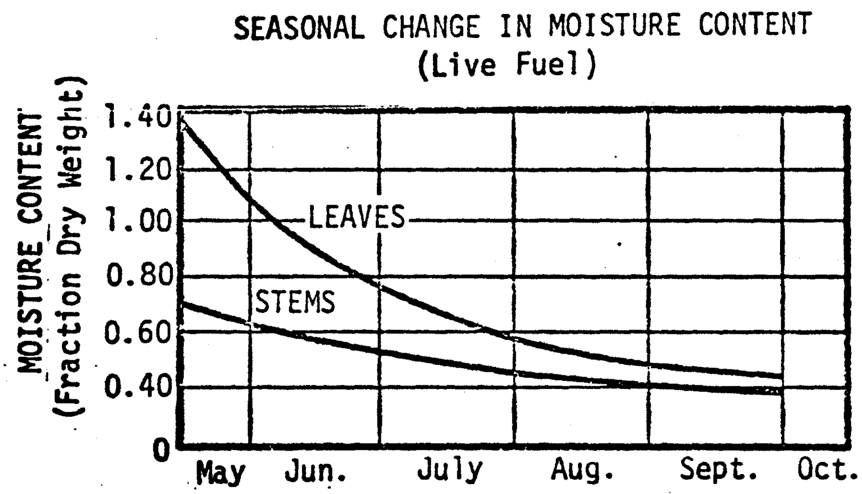


FIGURE 35 Seasonal Change in Moisture Content of Live Fuel from May Through October.

These data, together with extrapolated data for a wet year, are shown in Table 4.

Table 4. Southern California Rainfall

CONDITIONS	RAINFALL	PERIOD OF RAINFALL
Wet Year	167 cm	Sept. - June
Average Year	76 cm	Oct. - May
Dry Year	18 cm	Nov. - April

The above data are the essence of what has been found in the literature that pertains to the moisture content of live chaparral. These data will be used as the basis for a preliminary model of live fuel moisture content.

It is desired that the model be capable of describing the seasonal (from May through September) live fuel moisture content as a function of rainfall during the preceding winter, and as a function of fuel size class. As a first step, fit equations to the curves of Figure 35. The equation

$$M_{FL}(t,S_i) = L_o(S_i)\ \varepsilon^{-\alpha_{S_i}(t-t_o)} \qquad (1)$$

where

S_i	=	fuel size class
	=	1, leaves
	=	2, stems < 0.25 in. dia.
	=	3, 0.25 $\leq$ stems $\leq$ 0.5 in. dia.
	=	4, 0.5 $\leq$ stems < 1.0 in. dia.
	=	5, 1.0 $\leq$ stems < 3.0 in. dia.

M_{FL} = live fuel moisture fraction
L_o = live fuel moisture fraction at start of May
$L_o(1)$ = 1.38
$L_o(2)$ = 1.04
$L_o(3)$ = 0.94
$L_o(4)$ = 0.80
$L_o(5)$ = 0.70
α_i = live fuel drying rate
α_1 = 0.23
α_2 = 0.173
α_3 = 0.115
α_4 = o.086
α_5 = 0.057.

The amount of rainfall during the winter may affect live fuel moisture in two ways: (1) the different amount of available water may change initial fuel moisture; and (2) the last rainfall may occur at a different time, thus also affecting fuel moisture on 1 May. These effects are incorporated by assuming that: Figure 35 represents an average year; due to runoff only twenty percent of the difference in rainfall from the average level (76 cm) affects initial moisture content $(l_o(S_i))$; and initial time t_o is the date of last rain fall in the preceding winter.

Incorporation of these modifications into (1) yields

$$M_{FL}(t,S_i) = \left(1+\log\left(\frac{D_{TR}}{76}\right)\right) L_o(S_i)\ \varepsilon^{-\alpha_{S_i}(t-T_{LR})} \qquad (2)$$

where

D_{TR} = depth of total rainfall in previous winter
T_{LR} = time (month and fraction) of last rainfall.

Curves of seasonal live fuel moisture content from May through September, and two fuel-size classes are shown in Figure 36 for a wet winter, an average (rainfall) winter, and a dry winter.

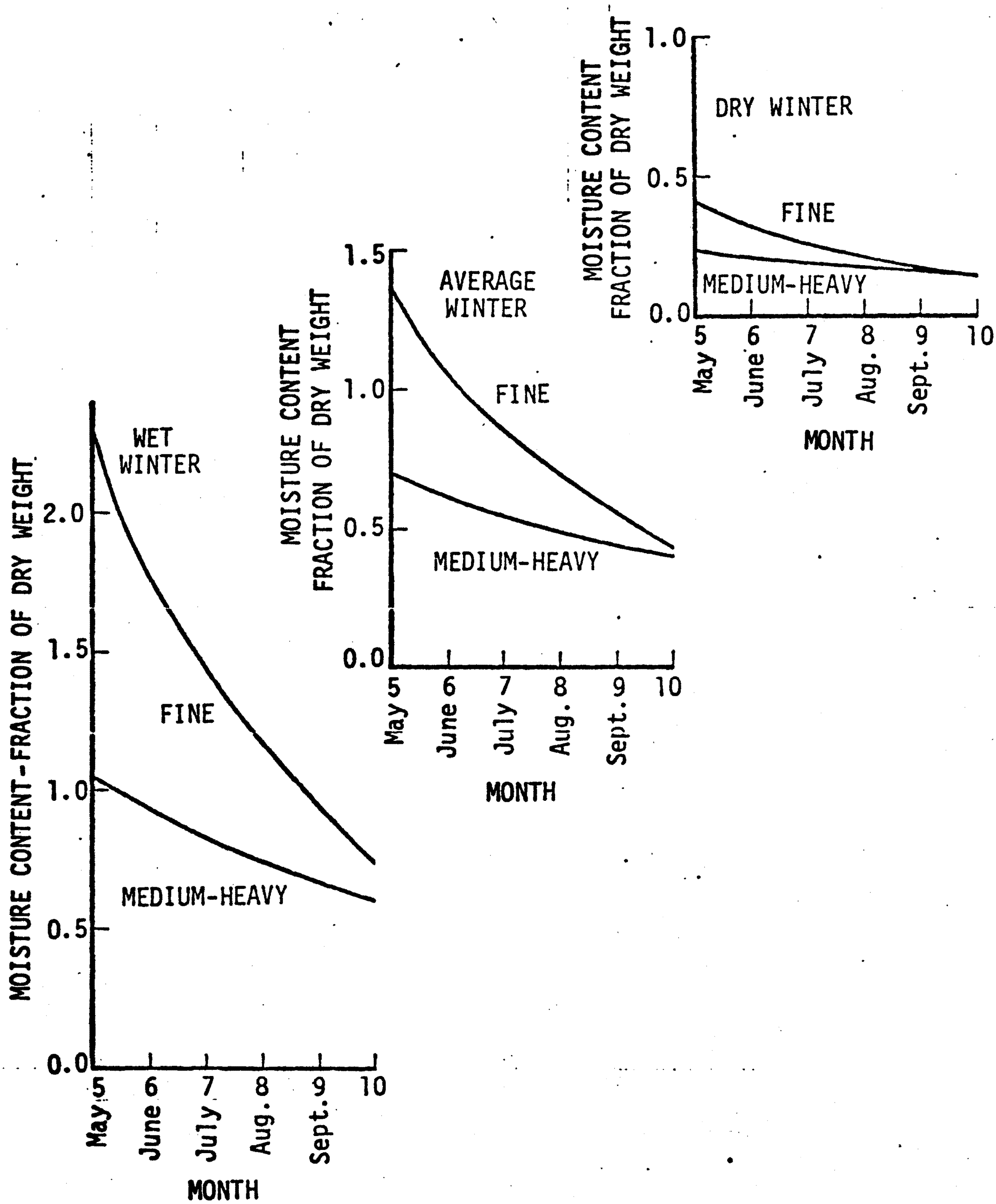


FIGURE 36 Seasonal Live Fuel Moisture Content as a Function of Rainfall and Fuel Size Class

26.1 FMOISL Version 1.00 17 December 1975

1. Purpose

This routine estimates moisture fraction for five size classes of live fuels.

2. Arguments

<u>OUTPUT</u>

EMF(2,N) = Live fuel moisture (fraction of oven-dry weight)

3. Procedure

Live fuel moisture fraction ($M_f(2,i)$) is estimated by:

$$M_f(2,i) = (1 + \log(D_{TR}/76))L_i \, d^{-\alpha_i(t-T_{LR})}$$

where

D_{TR} = depth of total rainfall in previous winter
T_{LR} = time (month and fraction) of last rainfall
α_i = drying rate per size class
L_i = initial moisture fraction per size class.

4. Model Results

Figure 37 shows live fuel moisture fraction plotted versus days since 1 May for five size classes. Figure 38 shows the effect of total rainfall and time of last rainfall upon live fuel moisture fraction, reaction intensity and rate of spread in mixed chaparral (TYP=5), of 25 years age (FAGE), for a relative humidity of 30 percent (HUM), a temperature of 65°F (TMP), and a wind speed of 268 m/min (WINDS). It can be seen that M_f increases monotonically for later dates of last rainfall (LRF) and amount of total rainfall (TRF), concomitantly, both reaction intensity and rate of spread are monotonic decreasing functions of the same variables.

Figure 39 shows live fuel moisture fraction and reaction intensity for two values of relative humidity and temperature plotted against day of year for the indicated conditions. Live fuel moisture decreases monotonically with day of year, and reaction intensity exhibits a monotonic increase. The reaction intensity curves indicate the sensitivity of wildland fire intensity to local meteorological conditions.

FIGURE 37 LIVE FUEL MOISTURE CONTENT AS A FUNCTION OF FUEL SIZE CLASS AND DAYS SINCE 1 MAY

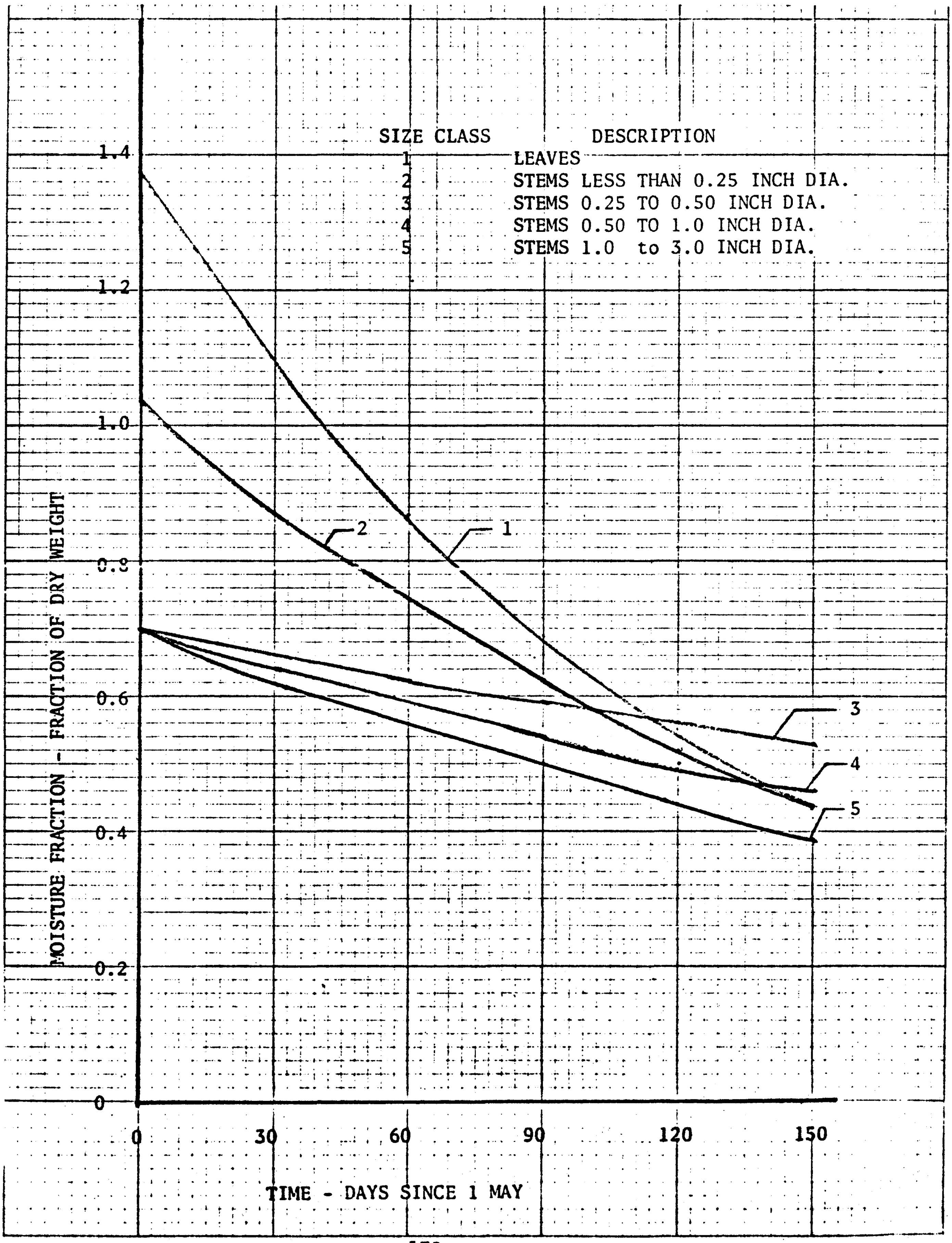


Figure 38. Effect of Total Rainfall and Time of Last Rainfall Upon Live Fuel Moisture

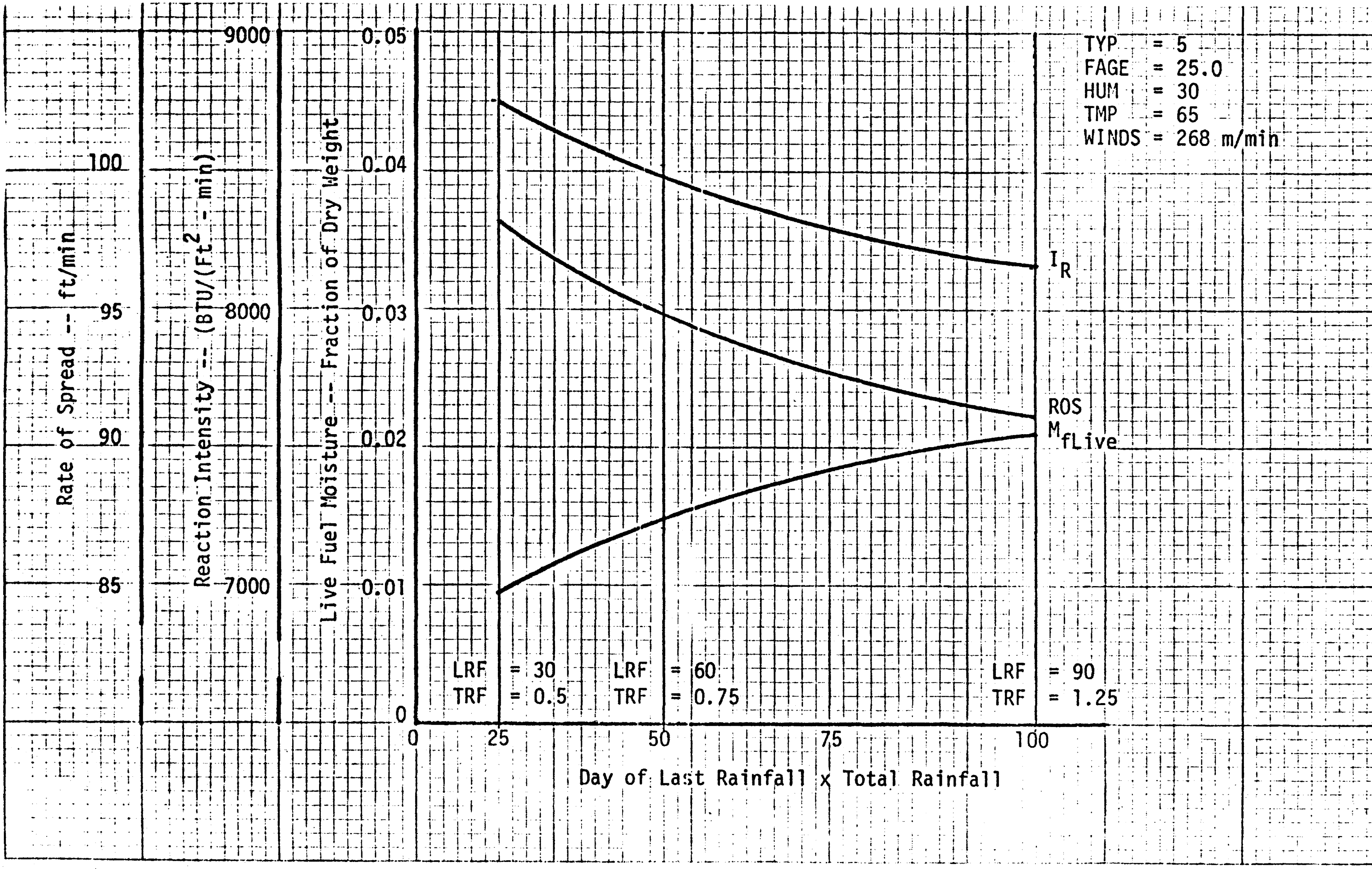
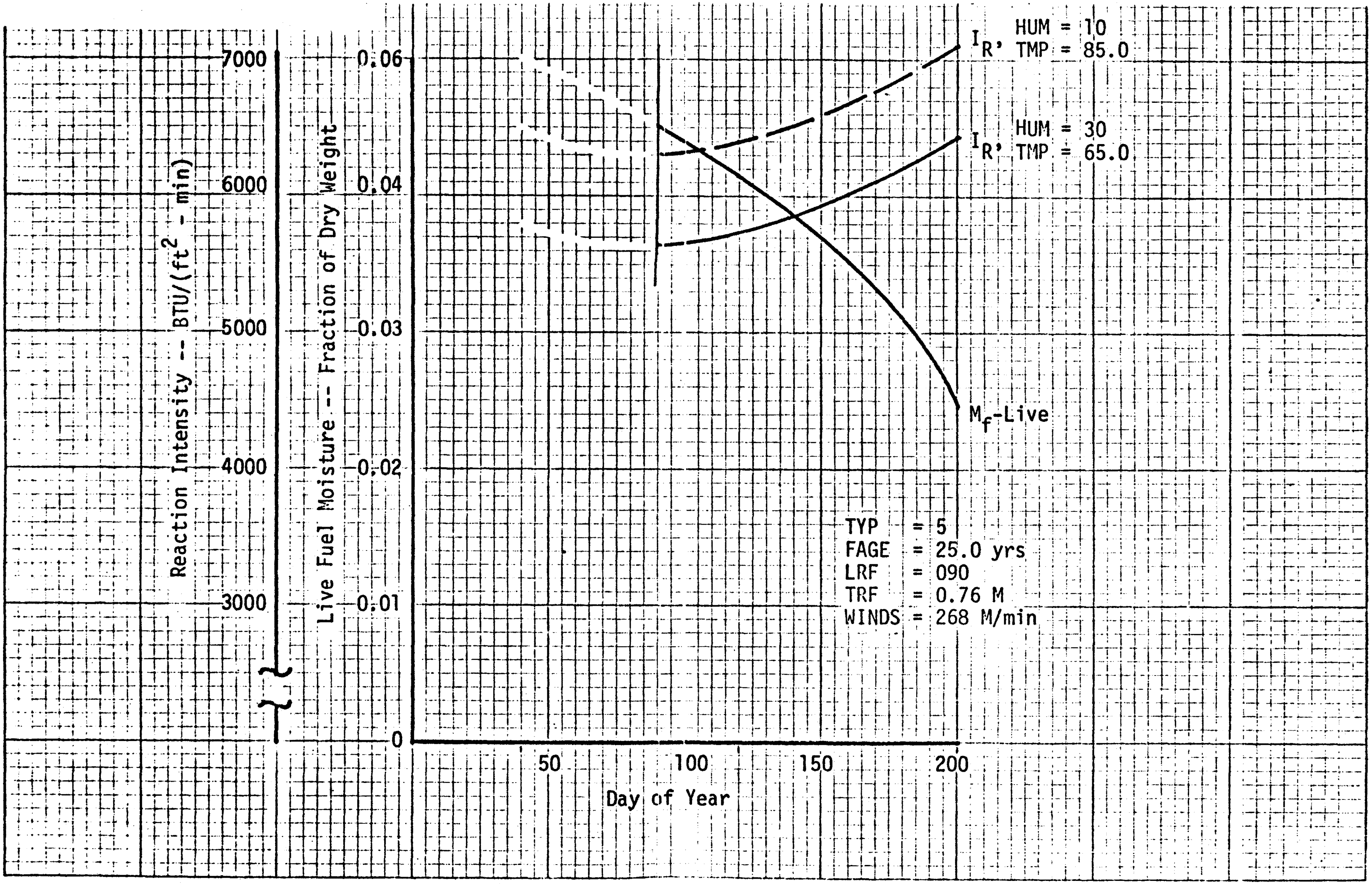


Figure 39. Live Fuel Moisture Fraction as a Function of Day of Year and Reaction Intensity as a Function of Day of Year for Two Relative Humidity-Temperature Conditions



```
      SUBROUTINE FMOISL(JDAY,IYR,N,EMF)
C
C     THIS ROUTINE DETERMINES MOISTURE FRACTION
C     FOR FIVE SIZE CLASSES OF LIVE FUEL
C     VERSION 1.00    1/12/76    SANDERLIN
C
C         INPUTS FROM LIST
C     JDAY    - DAY OF DESIRED FUEL CHARACTERISTICS (DAYS)
C     IYR     - YEAR OF DESIRED FUEL CHARACTERISTICS (YEARS-1900)
C     N       - AN ARRAY OF NUMBER OF CLASSES PER CATAGORY (LENGTH=2)
C         INPUTS FROM COMMON /SPLMET/
C     IYLR    - YEAR OF LAST RAINFALL (YEARS-1900)
C     JDLRF   - DAY OF LAST RAINFALL (DAYS)
C     DRAIN   - DEPTH OF TOTAL RAINFALL (METERS )
C         OUTPUT TO LIST
C     EMF(2,K)-(K=1,N(2))-LIVE FUEL MOISTURE (FRACTION OF DRY
C                             WEIGHT)
C
C     ROUTINES USED
C          MOFRAC
C
      DIMENSION EMF(2,6),N(2),EMO(5),ALPHA(5)
      COMMON/SPLMET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
C          DEFINE EMO AND ALPHA
      DATA EMO/0.0345,0.101,0.338,0.60,0.824/
      DATA ALPHA/0.23,0.173,0.115,0.086,0.057/
C
C     SAVE AN INPUT FOR USE IN DO LOOP
C
      ND02=N(2)
C
C     DETERMINE SEASON FOR THIS DAY
C
      TDIFF=0.0
      IF(JDAY .LT. 120  ) GO TO 10
C          DRY SEASON, CONVERT DAY OF YEAR
C          TO MONTH PLUS FRACTION.
      CALL MOFRAC(IYR,JDAY,T)
      CALL MOFRAC(IYLR,JDLRF,TLRF)
      TDIFF=T-TLRF
C
C     COMPUTE MOISTURE FRACTION FOR N(2) SIZE
C     CLASSES OF LIVE (M=2) FUELS
C
   10 CONTINUE
      FACTOR=1.0+ALOG10(DRAIN/0.76)
      DO 20 J=1,ND02
      EMF(2,J)=FACTOR*EMO(J)*EXP(-ALPHA(J)*TDIFF)
   20 CONTINUE
      RETURN
      END
```

27.0 MOISTURE CONTENT OF DEAD FUEL

The moisture content of dead chamise is directly dependent on atmospheric moisture, since dead fuels are hygroscopic. Because much of the dead fuel in chamise is small and the fuel bed is very porous, the moisture content of the dead fuel can be expected to respond very quickly to changes in relative humidity and temperature. Since dead fuel significantly affects the way chamise burns, the behavior of fire in chamise can also be expected to change quickly with variations in temperature, humidity and wind speed. Observations of wildland fires in chamise tend to bear this out (Reference 8).

If a piece of hygroscopic (dead) fuel of given moisture content is exposed to air of constant and markedly different moisture content, the moisture content of the fuel will change. The change will be rapid at first, then slows, and finally stops. The moisture content of the fuel is then in equilibrium. When exposed to saturated air the fuel reaches the highest moisture content possible by absorption of water vapor from the air. This moisture content is the fiber saturation moisture content of the fuel. The fiber saturation moisture is known to vary for different kinds of fuels, but is usually about 30 percent of the dry weight of the fuel.[9] Data are given in Reference 9 on equilibrium moisture content as a function of temperature and relative humidity. An equation fit to these data yields

$$M_S = 0.25h_\gamma \varepsilon^{-0.002(T-40)} - 10^{-3}\left\{0.9 + 0.1\ \sin[2.42(T-40)]\right\} h_\gamma^{\,2} \qquad (3)$$

where

M_S = saturation moisture content - percent of dry weight
T = Free stream temperature - degrees F
h_γ = relative humidity - percent.

A plot of (3) and data from Reference 9 are shown in Figure 40.

The behavior of fuel moisture content as a function of time is a function of the initial fuel moisture content, humidity, temperature and wind speed. An expression of this behavior is given by[10]

$$M_D(t) = M_o \varepsilon^{\gamma(t-t_o)} \quad (4)$$

where

$M_L(t)$ = moisture content (grams)
M_{Lo} = moisture content at t_o
r = drying rate constant
t_o = initial time (minutes)
t = time (minutes).

The drying rate constant is of the form[5]

$$r = 1.14 \times 10^{-6}\, S_w(p_{vi} - p_{vo})C_S \quad (5)$$

where

S_w = wind speed (cm/sec)
p_{vi} = vapor pressure in free stream
p_{vo} = vapor pressure at fuel surface
C_S = fuel size class coefficient.

Assume that the fuel moisture content has reached steady state with the ambient conditions at the initial time. The fuel moisture content at t_o is then the saturated moisture content for the fuel under ambient conditions at

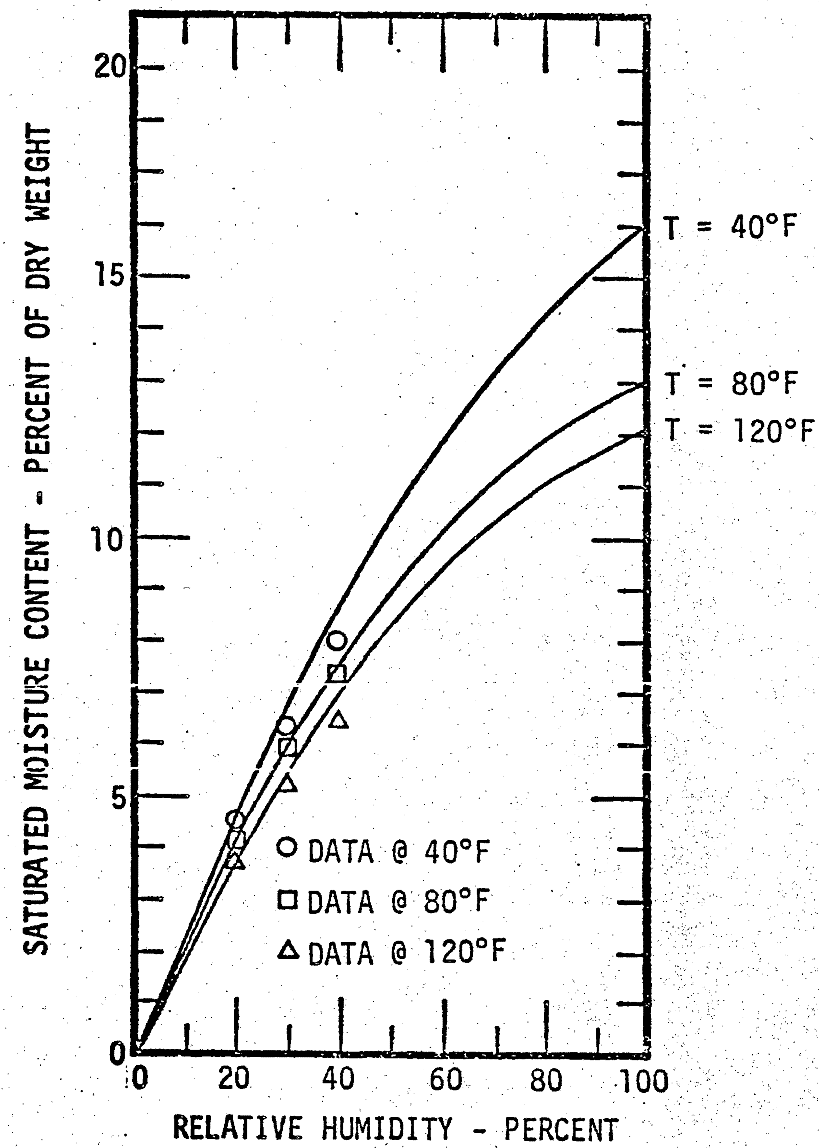


Figure 40 Saturated Moisture Content as a Function of Temperature and Humidity

that time. Thus p_{vi} may be considered as the saturated vapor pressure at the surface of the fuel. Vapor density, vapor pressure and relative humidity are related by

$$h_v = \rho_{vi}/\rho_{vo} - p_{vi}/p_{vo} \tag{6}$$

where

ρ_{vi}	=	free stream vapor density
ρ_{vo}	=	saturated vapor density at the mixture temperature.

Values for vapor pressure can be obtained from steam tables; however, a simple fit to these data yields

$$p_v(T) = 7.55\ \varepsilon^{0.0305(T-40)} \tag{7}$$

where

P_v	=	vapor pressure in mm Hg
T	=	temperature in degrees Farenheit.

As an approximation[10] write

$$\begin{aligned} p_{vi} &= p_v(T_d(h_\gamma)) \\ P_{vo} &= p_v(T_w(h_\gamma)) \end{aligned} \tag{8}$$

where

T_d	=	dry bulb temperature
T_w	=	wet bulb temperature.

Wet bulb temperature is given as a function of dry bulb temperatures and relative humidity by

$$T_w(h_r,T_d) = 35 + 0.162\ (h_r-10) + [0.58+S.1\varepsilon-3(h_r-10)](T_d-50), \qquad (9)$$

which fits the conversion data[10] with an error of less than about 2 percent.

The selection of size class coefficients is based on the expected response times of fine and large dead fuels,[8] which indicate that the response time of fine fuel should be a few minutes, while large limbs (3 in. diameter) may require up to 4 days to reach equilibrium. Time constants to be used with the three size classes are thus defined to be:

Size Class	Time Constant
1	τ_1 = 0.5 hours
2	τ_2 = 1.74 hours
3	τ_3 = 6.0 hours
4	τ_4 = 19.3 hours
5	τ_5 = 80.0 hours

The size class coefficients (C_S) in (5) are defined from the time constants and data of Reference 10 to be:

Size Class	Coefficient
1	C_1 = 1.0
2	C_2 = 0.29
3	C_3 = 0.083
4	C_4 = 0.026
5	C_5 = 0.00625

From (5), the drying rate from an initial state to a final state can be defined as shown in Table 5 for one initial condition and four final conditions.

In natural dead fuels, absorption as well as drying may occur, and the relative rates of absorption and drying determine the fuel moisture history. Assume two sets of conditions are specified; an initial condition and a final condition. The initial condition is specified by t_o, T_{do}, h_{ro}, S_{wo} and the final condition is specified by t_f, T_{df}, h_{rf}, and S_{wf}. It will be assumed that both absorption[11] and drying[10] are exponential processes with a limit. A general expression for an exponential process with a limit is

$$M(t) = A_o + A_1 e^{-rt}. \tag{10}$$

Require that

$M(t) = M_o$ at $t \quad t_o$

$M(t) = M_f$ for t very large.

and (10) can be written in the form

$$M(t) = M_f + (M_o - M_f)e^{-r(t-t_o)}. \tag{11}$$

A graphic form of (11) is shown in Figure 41, where it can be seen that the expression satisfies the following constraints:

drying occurs only if M_f is less than M_o;

absorption occurs only if M_f is greater than M_o;

both drying and absorption terminate as $M(t)$ approaches M_f.

Table 5. Drying Rates for Four Conditions

$$v = (1.14e\text{-}6)S_{w_f}(P_{V_F} - P_{0_1})C_S$$

INITIAL STATE	(1) MILD FINAL STATE	(2) MODERATE FINAL STATE
T_d = 80°F	T_d = 60°F	T_d = 80°F
h_v - 25%	h_r = 50%	h_r = 30%
S_W = 894 cm/sec	S_W = 894 cm/sec	S_W = 44.7
T_W = 57.1°F	T_W = 48.9°F	T_W = 58.7°F
P_{V1} = 25.6 mm Hg	P_{V1} = 13.8 mm Hg	P_{V1} = 25.6
P_{VO} = 12.7 mm Hg	P_{VO} = 10.0 mm Hg	P_{VO} = 13.4
M_S = 5.14%	M_S = 9.57%	M_S = 6.02%
	r_f = 1.12 E-3	r_f = 6.60 E-4
	r_m = 9.3 E-5	r_m = 5.46 E-5
	r_h = 6.72 E-6	r_h = 3.94 E-6
	(3) SEVERE FINAL STATE	**(4) VERY SEVERE FINAL STATE**
	T_d = 90°F	T_d = 110°F
	h_r = 10%	h_r = 2%
	S_W = 2235 cm/sec	S_W = 3129 cm/sec
	T_W = 58.2°F	T_W = 66.1°F
	P_{V1} = 34.7	P_{V1} = 63.9
	P_{VO} = 13.2	P_{VO} = 16.8
	M_S = 2.16%	M_S = .431
	r_f = 5.60 E-2	r_f = 1.83 E-1
	r_m = 4.65 E-3	r_m = 1.52 E-2
	r_n = 3.36 E-4	r_n = 1.09 E-3

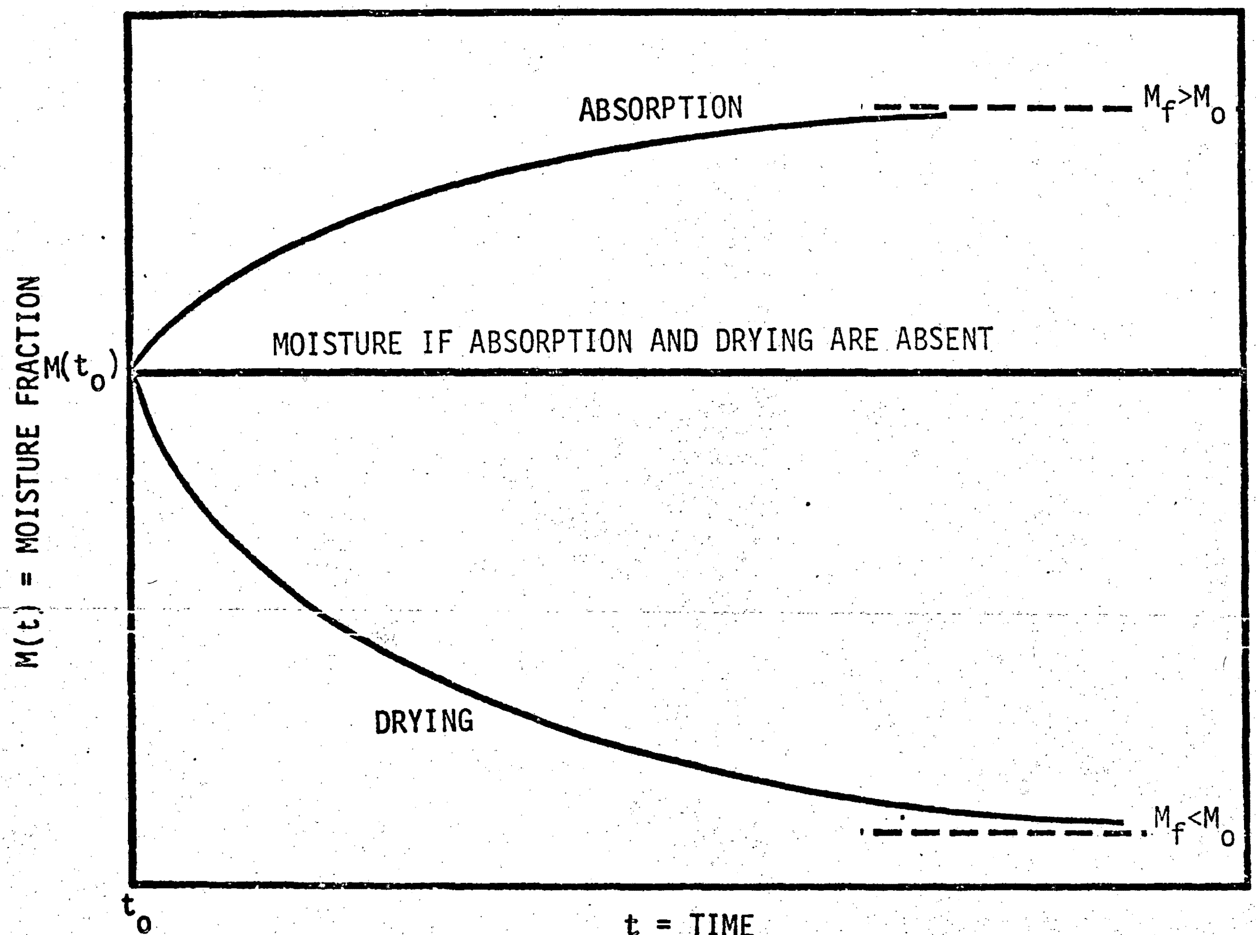


Figure 41. Moisture Fraction Model

Although (11) is a hueristic model, it provides results that are intuitively satisfying. The remaining element of (11) to be defined is the rate (r). Intuitively one would expect the rate of either drying or absorption to be a function of the difference in the initial and final moisture contents, the free-stream air speed and fuel particle size. The rate expression (5) satisfies these conditions (at least for drying) and will thus be used to define the rates of both drying and absorption.

1. Purpose

This routine estimates moisture fraction for 5 size classes of dead fuels.

2. Arguments

<u>INPUT</u>

TEMP	=	local temperature (deg. C)
HREL	=	local relative humidity (percent)
WINDS	=	local wind speed (m/min)
N	=	number of size classes

<u>OUTPUT</u>

EMF(1,N)	=	dead fuel moisture content (fraction of oven-dry weight)

3. Procedure

Dead fuel moisture fraction is estimated from:

$$M_f(1,i) = M_{F_i} + (M_{o_i} - M_{F_i})\, e^{-r_i(t-t_o)}$$

where

$M_f(1,i)$	=	moisture fraction of dead fuels of size class i over the interval (t_o, t_F)
M_{F_i}	=	moisture fraction of dead fuels of size class i at $t = t_F$.
M_{o_i}	=	moisture fraction of dead fuels of size class i at $t = t_o$

r_i = rate of change of moisture fraction with respect to time for dead fuels of size class i.

4. MODEL RESULTS

Figure 42 shows equilibrium dead fuel moisture fraction plotted against a function of relative humidity and temperature for the set of indicated conditions: TYP=5, chaparral, FAGE = fuel age, LRF = Julian date of last rainfall, TRF = total rainfall (m) during preceding winter, DOY = day of year. It can be seen from Figure 42 that equilibrium dead fuel moisture fraction increases monotonically with increasing relative humidity and with decreasing temperature.

Figure 43 shows the effect of drying upon dead fuel moisture fraction and the attendant reaction intensity. It can be seen that the drying rate is modeled as an exponential with a limit, which is a very sensitive function of local wind speed. As is to be expected, a monotonic decrease in dead fuel moisture fraction results in a monotonic increase in reaction intensity.

Figure 42 Effect of Humidity-Temperature Upon Dead Fuel Equilibrium Moisture Fraction

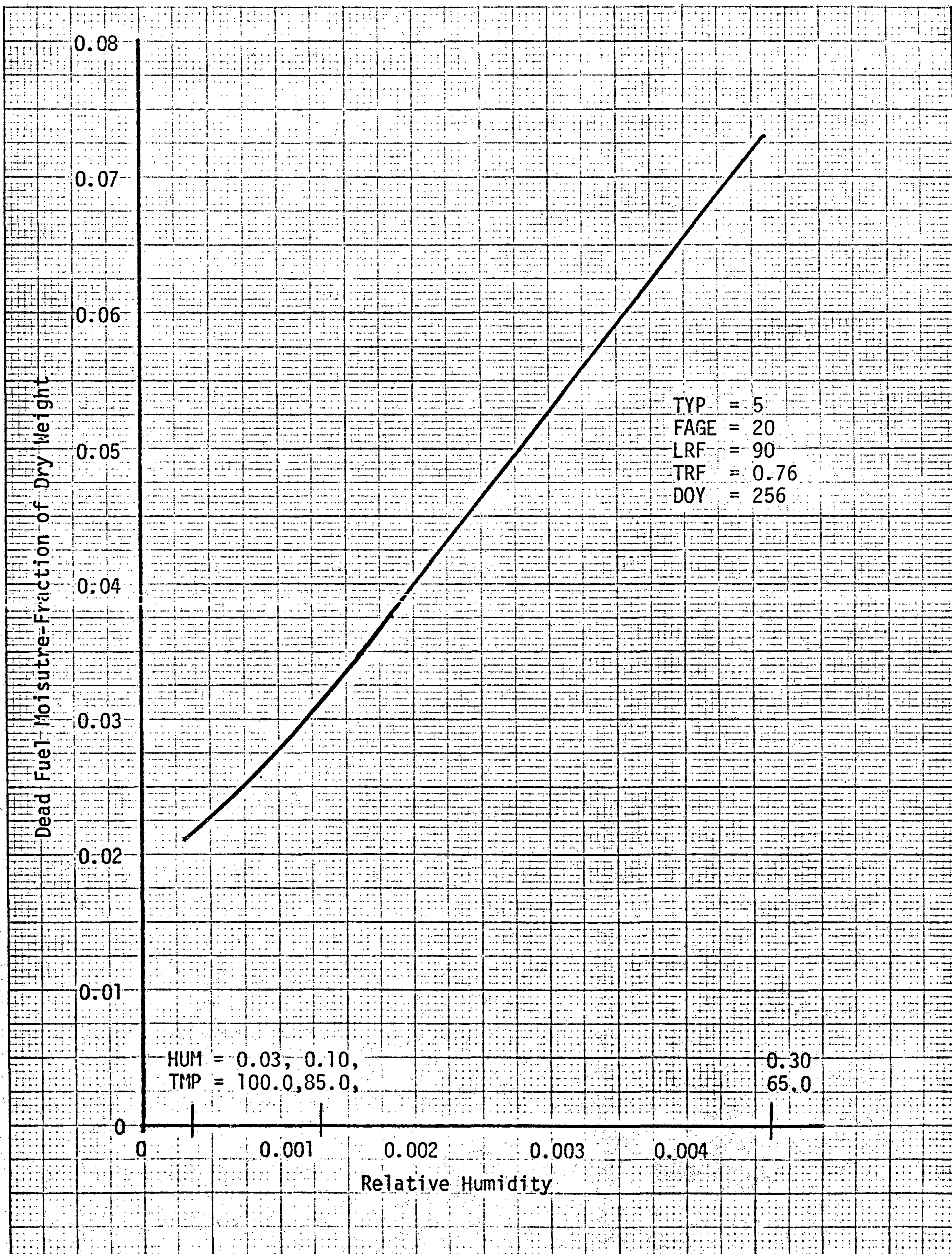
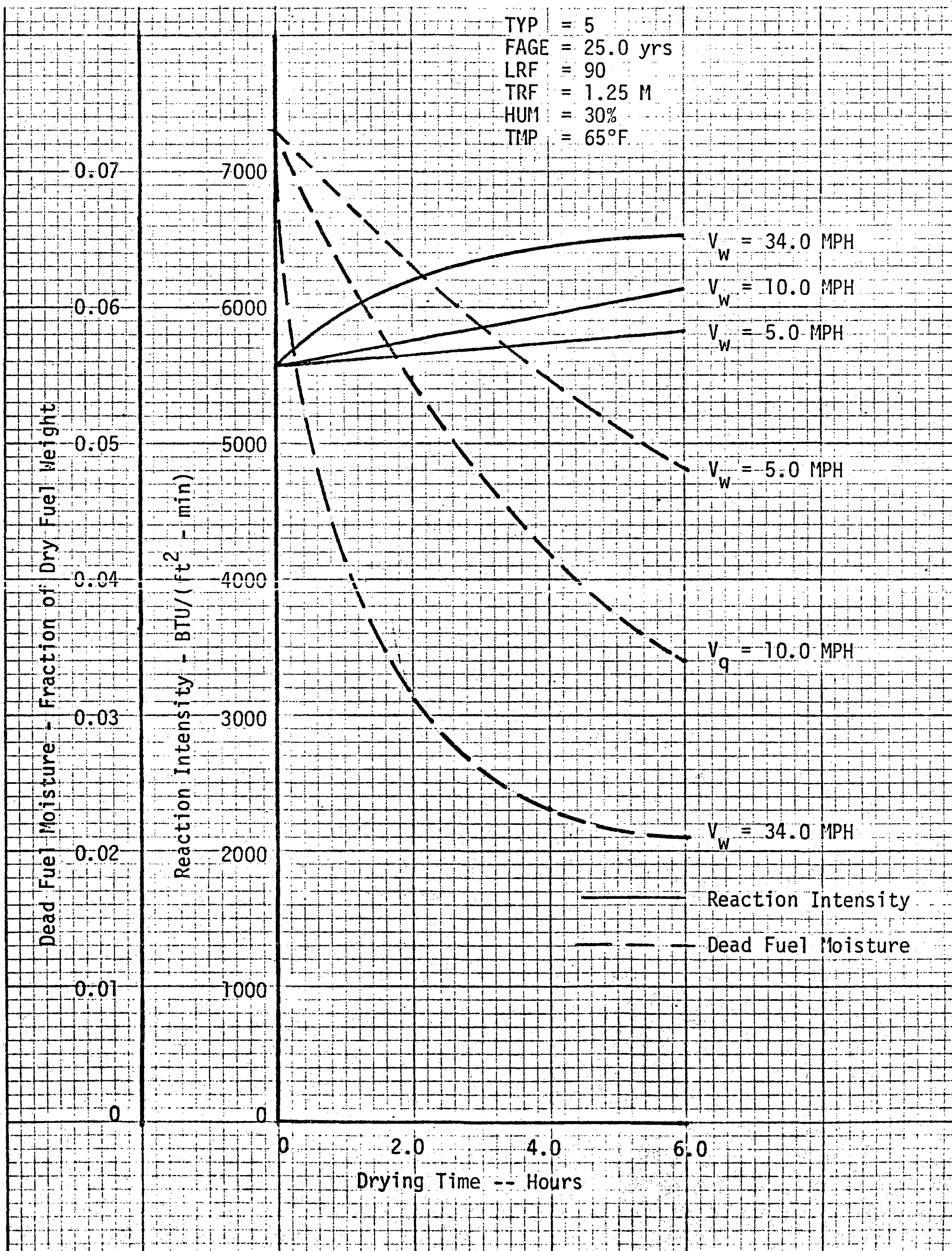


Figure 43 Effect of Drying Upon Dead Fuel Moisture Fraction and Reaction Intensity



```
      SUBROUTINE FMOISD(TDOC,TEMP,HREL,WINDS,N,EMF)
C
C     THIS ROUTINE DETERMINES MOISTURE FRACTION OF
C     FIVE DEAD FUEL SIZE CLASSES.
C     VERSION 1.00      1/12/76    SANDERLIN
C
C         INPUTS FROM LIST
C     TDOC    - TIME OF CALCULATION (MINUTES FROM MIDNIGHT)
C     TEMP    - TEMPERATURE (DEG. C)
C     HREL    - RELATIVE HUMIDITY (FRACTION)
C     WINDS   - WINDSPEED (METERS/MIN.)
C     N       - AN ARRAY OF THE NUMBER OF SIZE CLASSES PER
C               CATAGORY (LENGTH = 2)
C         INPUT FROM COMMON /SPLMET/
C     TDOM    - TIME OF DAY OF MET. DATA COLLECTION (MINUTES
C               FROM MIDNIGHT)
C         OUTPUT TO LIST
C     EMF(1,K)-(K=1,N(1))- DEAD FUEL MOISTURE (FRACTION OF DRY
C                             WEIGHT)
C
C     ROUTINES USED
C          FUNCTION EQMS(TEMP,HREL)
C          SUBROUTINE DRYRAT(HREL,TEMP,WINDS,N,RD)
C
C     DIMENSION VARIABLES AND DEFINE COMMON BLOCKS
C
      DIMENSION EMF(2,6),N(2),RD(5),EMFMN(1,5)
      COMMON/SPLMET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
      DATA EMFMN/0.01,0.02,0.06,0.12,0.12/
C
C     SAVE DO LOOP PARAMETER
C
      NDOI=N(1)
C
C     COMPUTE INTERVAL BETWEEN MET DATA TIME AND
C     CALCULATION TIME. NOTE= INTERVAL IS ASSUMED
C     TO BE NO GREATER THAN 24 HOURS.
C
      TDEL=TDOC-TDOM
      IF(TDEL .LT. 0.0)TDEL=TDEL+1440.0
C
C     DEAD FUEL EQUILIBRIUM MOISTURE CONTENT AT TMET
C
      EMS=EQMS(TEMP,HREL)
C
C     DEAD FUEL DRYING RATE COEFFICIENTS
C
      CALL DRYRAT(HREL,TEMP,WINDS,N,RD)
C
C     DEAD FUEL MOISTURE FRACTION
C
      DO 10 J=1,NDOI
      EMF(1,J)=EMS
      IF(EMF(1,J) .LE. EMFMN(1,J))EMF(1,J)=EMFMN(1,J)
   10 CONTINUE
      RETURN
      END
```

28.0 MONTH AND FRACTION MOFRAC Version 1.00 10 December 1975

1. PURPOSE

This routine transforms Julian date into number of the month and fraction.

2. ARGUMENTS

<u>INPUT</u>

JDOY - Julian day of year

<u>OUTPUT</u>

EMAF - Month and fraction

3. PROCEDURE

The days of each month are successively subtracted from the Julian date until a negative result is obtained. The preceding month is then the required integral month and the remainder of days divided by the number of days in the last subtracted month is the required fractional month.

4. COMMENTS

Required by the live fuel moisture routine FMOISL.

5. Flow Chart

Not required.

```
      SUBROUTINE MOFRAC(IYR,JDOY,EMAF)
C
C     THIS ROUTINE CALCULATES MONTH AND FRACTION
C     VERSION 2.00      1/9/76      SANDERLIN
C
      DIMENSION IDTAB(12)
C
      DATA IDTAB/31,28,31,30,31,30,31,31,30,31,30,31/
C
      IDAYS=JDOY
      IDTAB(2)=28
      IF(MOD(IYR,4) .EQ. 0) IDTAB(2) = 29
C
      DO 20 K=1,12
      MONTH=K
      IDAYS=IDAYS-IDTAB(K)
      IF(IDAYS .LE. 0) GO TO 25
   20 CONTINUE
C
   25 CONTINUE
      IDAYS=IDAYS+IDTAB(MONTH)
      AMON=FLOAT(MONTH-1)
      EMDAYS=FLOAT(IDTAB(MONTH))
      EMAF=AMON+FLOAT(IDAYS)/EMDAYS
      RETURN
      END
```

29.0 EQUILIBRIUM MOISTURE FRACTION (EQMS) Version 1.0 17 December 1975

1. Purpose

This function estimates dead fuel equilibrium moisture fraction for specified meteorological conditions.

2. Arguments

<u>INPUT</u>

TMPR	=	temperature (°C)
HREL	=	relative humidity (percent)

<u>OUTPUT</u>

EQMS	=	equilibrium moisture content (fraction of oven-dry weight).

3. Procedure

Equilibrium moisture fraction is estimated by:

$$M_S = (h_R/4)\ \exp(-0.002(T_D-40))$$

$$-0.60/h_R\ (0.9 + 0.1\ \sin(0.0422(T_D-40)))$$

where

M_S	=	equilibrium moisture fraction
h_R	=	relative humidity
T_D	=	dry bulb temperature

4. Comments

Equilibrium moisture content is employed in determining dead fuel moisture fraction.

5. Flow Chart

Not required.

```
      FUNCTION EQMS(TMPR,HRELI)
C
C     THIS FUNCTION COMPUTES THE DEAD FUEL EQUILIBRIUM MOISTURE CONTENT
C     VERSION 1.00      1/9/76      SANDERLIN
C
C
C        INPUTS
C     TMPR   - THE TEMPERATURE IN DEGREES CENTIGRADE
C     HRELI  - THE RELATIVE HUMIDITY AS A FRACTION
C        OUTPUT
C     EQMS   - THE EQUILIBRIUM MOISTURE CONTENT
C
C     CONVERT TEMPERATURE TO DEGREES FAHRENHEIT
C
      TF = TMPR*1.8 + 32.0
C
C     COMPUTE EQUILIBRIUM MOISTURE CONTENT
C
      HREL=HRELI*100.0
      TM40 = TF - 40.0
      EQMS=HREL*(.25*EXP(-.002*TM40)-.001*HREL
     &      *(.9+.1*SIN(.0422*TM40)))*0.01
C
      RETURN
      END
```

30.0 DRYING RATE (DRYRAT) Version 1.00 17 December 1975

1. Purpose

This routine generates a drying rate for each size class of dead fuel at a specified location.

2. Arguments

INPUT

HRO	=	relative humidity (percent)
TMP	=	temperature (deg. C)
WS	=	wind speed (m/min)
NCS	=	number of size classes

OUTPUT

RIA(NCS)	=	drying rate coefficient

3. Procedure

The drying rate is given by 10:

$$R_i = 1.14 \times 10^{-6} S_w (P_{V1} - P_{VO}) C_{S_i}$$

where

S_w	=	wind speed (cm/sec)
P_{V1}	=	$P_V(T_D)$, T_b = dry bulb temperature (°F)
P_{VO}	=	$P_V(T_W)$, T_w = wet bulb temperature (°F)
$P_V(T)$	=	7.55 exp (0.0305(T-40))
T_w	=	$35 + 0.162(H_R-10) + [0.58 + 0.0051(H_R-10)](T_D-50)$

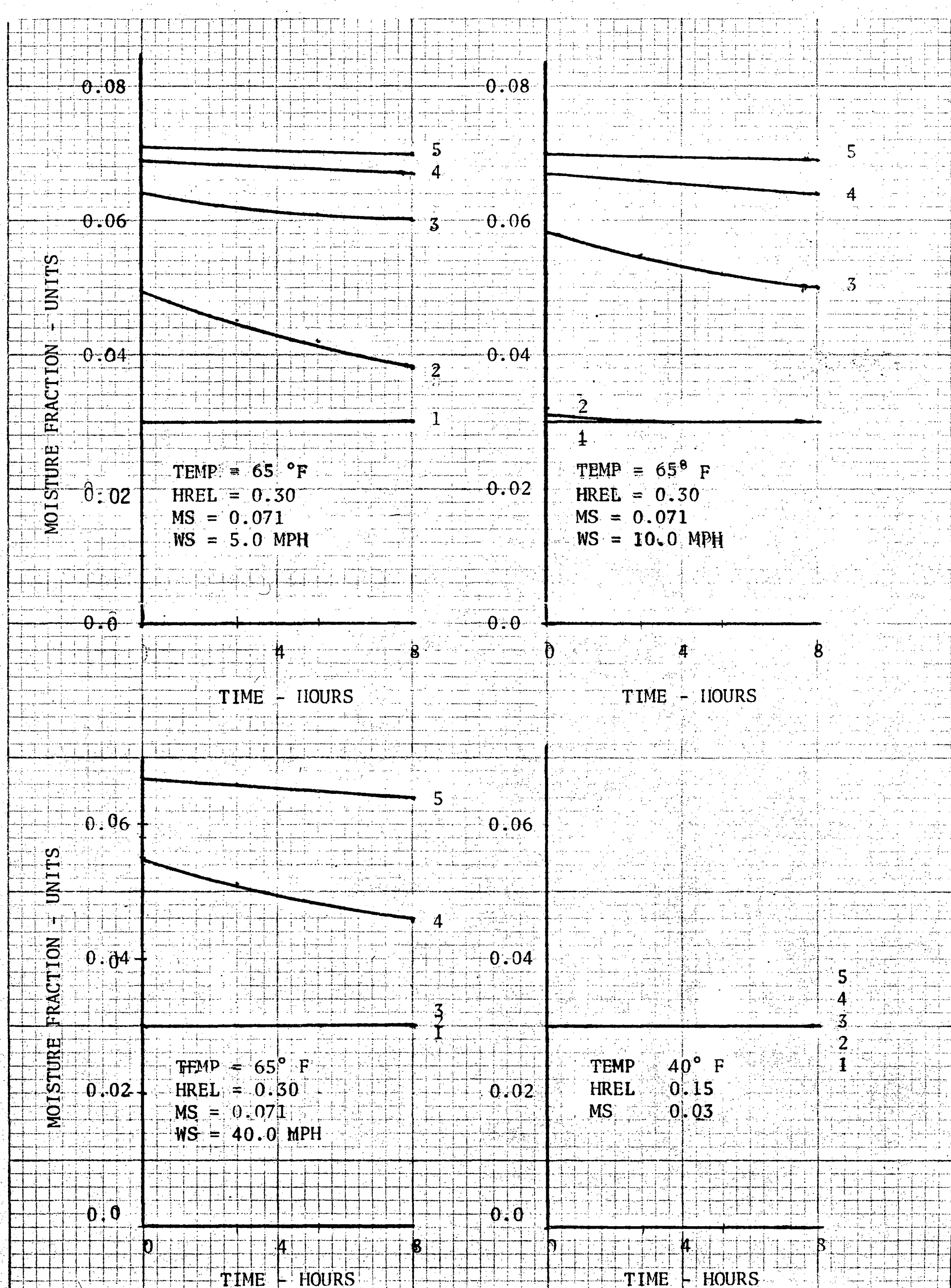
4. Model Results

Figure 44 presents dead fuel moisture fraction plotted against time for 5 fuel size classes and 4 sets of local meteorological conditions. It can be seen that drying rate is a sensitive function of wind speed. The equilibrium moisture fraction (MS) defines the maximum initial value of moisture fraction, and the minimum is set to 0.03. Thus, in the lower right-hand curve of Figure 44 the maximum and minimum values coincide such that no variation is permitted.

5. Flow Chart

Not required.

FIGURE 44 DEAD FUEL MOISTURE FRACTION



```
      SUBROUTINE DRYRAT(HRO,TMP,WS,NCS,RIA)
C     VERSION 1.00    1/9/76    SANDERLIN
C     THIS FUNCTION GENERATES A DRYING RATE FOR EACH
C     FUEL SIZE CLASS AT A SPECIFIED LOCATION
C
C     INPUTS (AT SPECIFIED LOCATION)
C          HRO = RELATIVE HUMIDITY AT TIME TODM
C                (FRACTION)
C          TMP = TEMPERATURE AT TIME TODM (DEG. C)
C          WS = WIND SPEED (METERS/MINUTE)
C     OUTPUTS
C          RIA = DRYING RATE COEFFICIENTS
C          FOR TIMES GREATER THAN TODM.
C
C          DIMENSION VARIABLES
      DIMENSION CS(5),RIA(5)
C          DEFINE SIZE COEFFICIENTS, CS
      DATA CS/1.0,0.29,0.083,0.026,0.00625/
C
C     DEFINE THE VAPOR PRESSURE FUNCTION
C
      PVAP(TF)=7.55*EXP(0.0305*(TF-40.0))
C
C     CONVERT TEMPERATURE TO DEG. F, WIND SPEED TO
C     CM/SEC.
C
      TDRY=TMP*1.8+32.0
      SWS=WS*1.666667
C
C     VAPOR PRESSURES
C
      PV1=PVAP(TDRY)
      HR10=100.0*HRO-10.0
      TWET=35.0+0.162*HR10+(0.56+5.1E-3*HR10)
     &      *(TDRY-50.0)
      PV0=PVAP(TWET)
C
C     DEAD FUEL DRYING RATE
C
      DO 10 J=1,5
      RIA(J)=-1.14E-6*SWS*(PV1-PV0)*CS(J)
   10 CONTINUE
      RETURN
      END
```

31.0 FUEL CHARACTERISTICS AT A POINT (FUELPT)

1. Purpose

This routine finds the fuel type and approximate land value for a specified point.

2. Arguments

Input

X	-	X coordinate of given point
Y	-	Y coordinate of given point

Output

ICLASS	-	Fuel classification index at a given point.
VALUE	-	Approximate land value at a given point.
AGE	-	Age of fuel (years)
IFTYPE	-	Fuel type index.

3. Procedure

This routine uses a two phase lookup of the fuel type. A table is accessed first for the fuel type. If a cell contains a negative fuel type, a search of lines (arbitrary areas) of fuels is made to determine whether or not the point is a portion of one line. If a point is not found to be in a line, it is assigned the absolute value of the tabular fuel type.

This particular algorithm was chosen because it allows the use of both tabular and non-tabular fuel data. A more efficient routine could be constructed if the tabular storage were doubled, and a second word kept to indicate where in the line-data the search is to begin. An algorithm of this type was not implemented, because the number of lines of fuel data is small for the prototype data base and storage conservation is currently more important than efficiency.

The use of this algorithm permits specification of line fuel characteristics along lines having widths less than a cell dimension. This is necessary to describe fuel modifications (e.g., line construction) which may have widths very much less than a cell dimension.

4. Comments

None.

5. Flow Chart

See Figure 45.

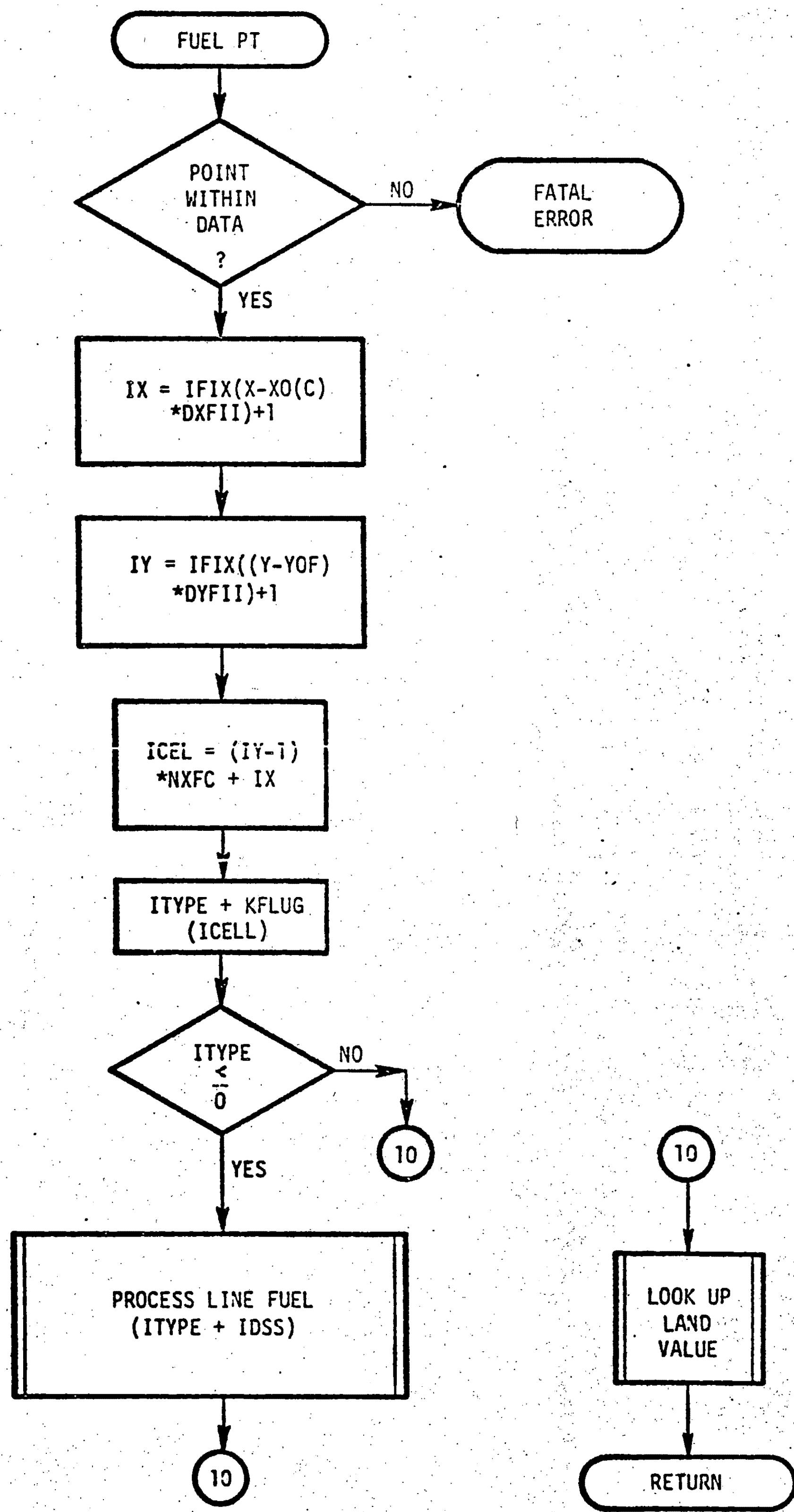


Figure 45 Logic Structure for Routine FUELPT

```
$$$$$SUBT1
      SUBROUTINE FUELPT(X,Y,ITYPE,VALUE,AGE,IFTYP)
C
C     THIS ROUTINE FINDS THE FUEL TYPE AND APPROXIMATE LAND VALUE
C     FOR A GIVEN POINT.
C     VER 2.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     X       - X COORDINATE OF A GIVEN POINT.
C     Y       - Y COORDINATE OF A GIVEN POINT.
C         OUTPUTS
C     ITYPE   - NOT USED
C     VALUE   - NOT USED
C     AGE     - AGE OF FUEL (YEARS)
C     IFTYPE  - TYPE OF FUEL INDEX
C
C         ROUTINES USED
C     FULVAL,ERRHED
      COMMON/FUEL2/KUADF,XOF,YOF,XMF,YMF,DXFI,DXFII,DYFI,DYFII,
     &             DXFO2,DYFO2,NXFC,NYFC,NCELF,AGET(50,50),IFTYPE
      COMMON /WARNS/ IWARN(3)
C
C       ****************
C
C     VERIFY POINT WITHIN LOADED BLOCK
C
    5 IF(XOF.LE.X .AND. X.LE.XMF
     &            .AND.
     &   YOF.LE.Y .AND. Y.LE.YMF)
     & GO TO 10
      XOUT=X*3.28084
      YOUT=Y*3.28084
      IF (IWARN(2).EQ.0)PRINT    2000,XOUT,YOUT
      IWARN(2)=1
 2000 FORMAT(3X,43H***WARNING***DATA BASE EXCEEDED-FUELPT-X,Y=,
     & 2E15.6)
C
C     CALCULATE GRID INDEX AND ACCESS TYPE
C
   10 IX=IFIX((X-XOF)*DXFII)+1
      IX=MIN0(IX,NXFC-1)
      IX=MAX0(IX,1)
      IY=IFIX((Y-YOF)*DYFII)+1
      IY=MIN0(IY,NYFC-1)
      IY=MAX0(IY,1)
C         GET FUEL TYPE
      IFTYP=IFTYPE
C         GET AGE
      AGE=AGET(IX,IY)
C         FILL UNUSED OUTPUTS
      ITYPE=IFTYPE
      VALUE=0.0
C
      RETURN
      END
```

32.0 FUEL LOAD (FULOD(IFTYP,AGE,DAY,FLOD)) Version 1.00 20 January 1976

1. Purpose

This routine estimates the total fuel load for use in FULCHR and elsewhere.

2. Arguments

<u>INPUTS</u>

IFTYP	=	Fuel type index
	=	1, short grass
	=	2, long grass
	=	3, brush (non-chaparral)
	=	4, chamise
	=	5, mixed chaparral
AGE	=	Fuel age (years)
DAY	=	Number of days since 1 May.

<u>OUTPUTS</u>

FULOD	=	Total fuel load (grams/meter^2)

3. Procedure

Total fuel load for the different fuel types is:

Short Grass

$$F_L = 83.0 \exp(0.00385 N_{day}) \tag{1}$$

F_L = fuel load - grams/meter^2

N_{day} = number of days since 1 May

Long Grass

$$F_L = 0.251 \exp(0.00385 N_{day}) \tag{2}$$

Brush

$$F_L = 0.325\, A_F/(4.03 + 0.806\, A_F) \tag{3}$$

A_F = fuel age - years

Chamise

$$F_L = 0.224A_F/(1.446 + 0.0315A_F) \quad (4)$$

Mixed Chaparral

$$F_L = 0.224A_F/(0.485 + 0.017A_F) \quad (5)$$

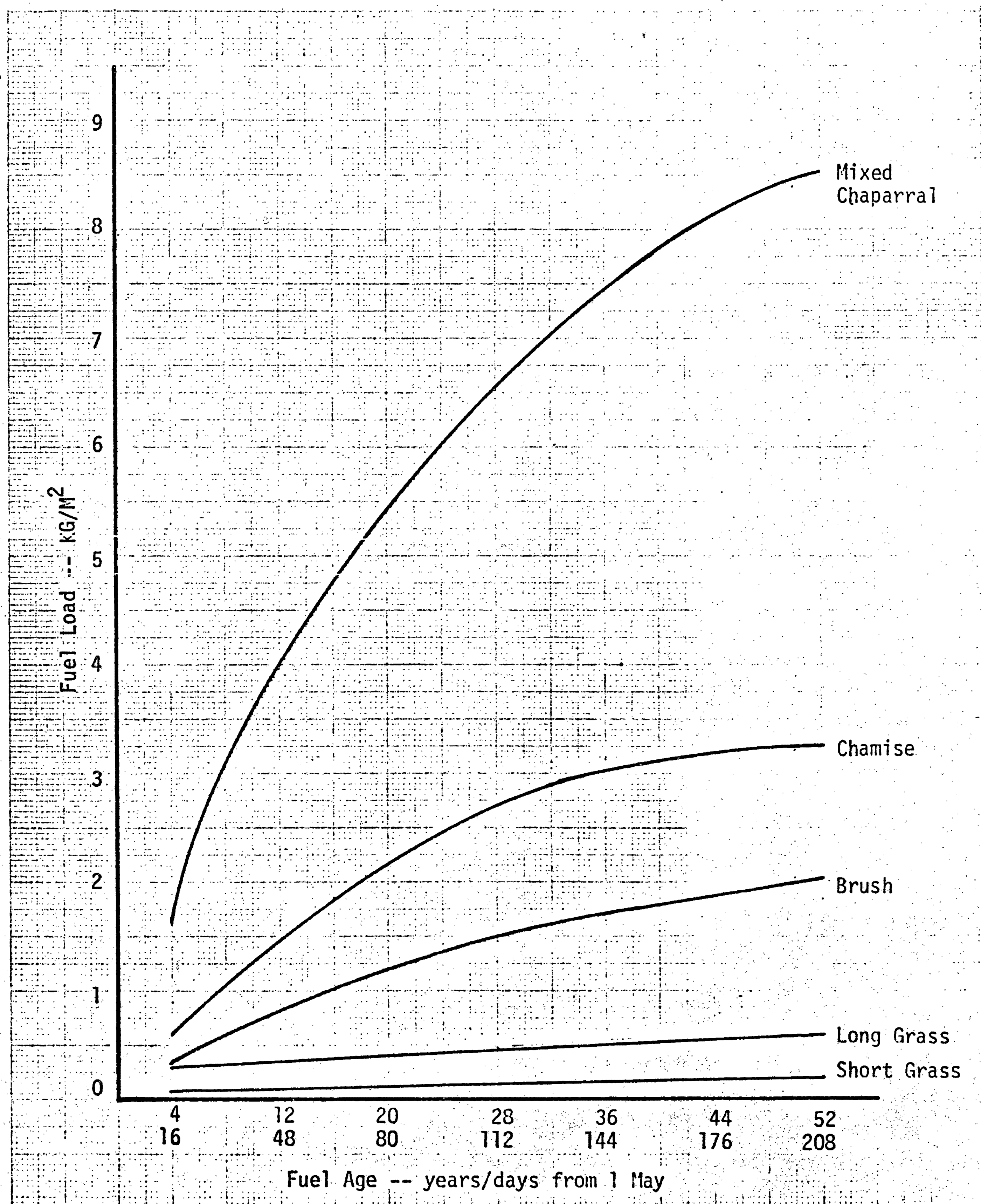
4. Model Results

Figure 45A shows fuel load plotted versus fuel age for the 5 fuel classes currently within the capability of the model. The abscissa has two scales: (1) fuel age in years is used for mixed chaparral, chamise and brush; and (2) days from 1 May are used for short and long grass.

5. Flow Chart

Not required.

Figure 45A Total Fuel Load as a Function of Fuel Type and Age



```
      SUBROUTINE FULOD(IFTYP,AGE,DAY,FLOD)
C     VERSION 1.0      1/20/76
C
C     PURPOSE=
C       THIS PROGRAM DETERMINES TOTAL FUEL LOAD
C       GIVEN FUEL TYPE AND AGE
C     ARGUMENTS=
C       INPUT
C          IFTYPE=1, SHORT GRASS
C                =2, LONG GRASS
C                =3, BRUSH(NON-CHAPARRAL)
C                =4, CHAMISE(PURE STAND)
C                =5, MIXED CHAPARRAL
C          AGE   =   FUEL AGE(YEARS)
C          DAY   =   DAY OF FIRE (DAYS SINCE MAY 1)
C       OUTPUT
C          FLOD  =   TOTAL FUEL LOAD(KILOGRAMS/METERS**2)
C
C       BRANCH ON FUEL TYPE INDEX
      GO TO(10,20,30,40,50),IFTYP
C
   10 FLOD=0.083*EXP(0.00385*DAY)
      GO TO 1000
C
C       FUEL TYPE IS LONG GRASS
   20 FLOD=0.251*EXP(0.00385*DAY)
      GO TO 1000
C
C       FUEL TYPE IS BRUSH
   30 FLOD=0.325*AGE/(4.03+0.0806*AGE)
      GO TO 1000
C
C       FUEL TYPE IS CHAMISE
   40 FLOD=0.224*AGE/(1.446+0.0315*AGE)
      GO TO 1000
C
C       FUEL TYPE IS MIXED CHAPARRAL
   50 FLOD=0.224*AGE/(0.485+0.017*AGE)
      GO TO 1000
 1000 CONTINUE
      RETURN
      END
```

33.0 METEOROLOGICAL DATA AT A POINT (PTMET)

1. Purpose

This routine returns meteorological data for a specified point.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

Two procedures are available: (1) for geostrophic data a series of special routines are called, and (2) for stored distribution data a double linear interpolation is employed.

4. Comments

None.

5. Flow Chart

See Figure 46.

Figure 46 Meteorological Data at a Point

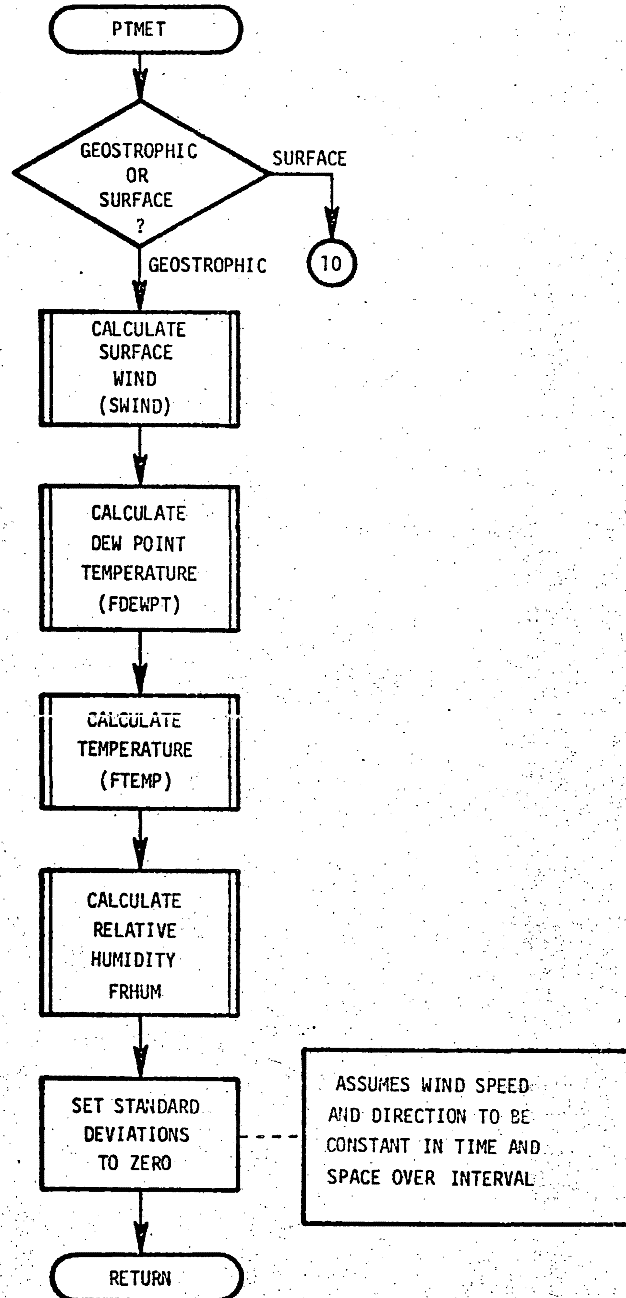
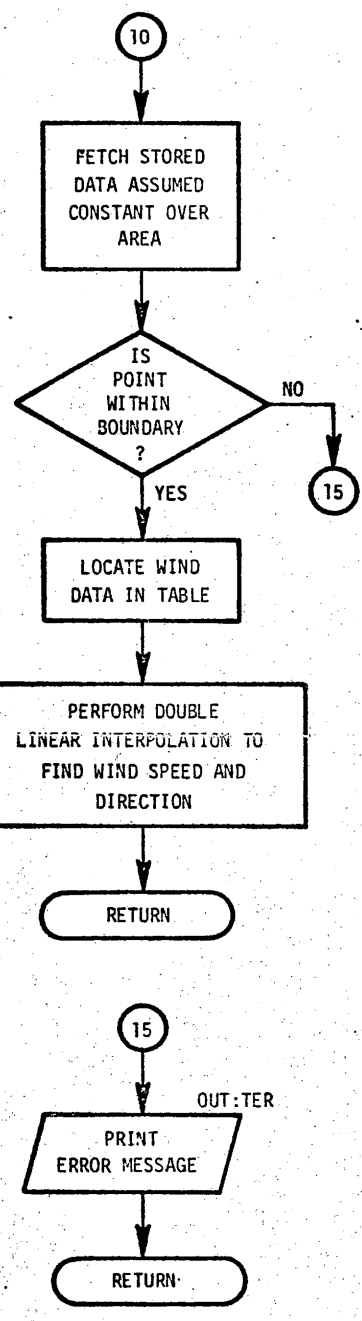


Figure 46 Meteorological Data at a Point (Cont'd)



```
      SUBROUTINE PIMET(X,Y,TIMEI,IDAY,WAS,WVEL,SDWAS,SDWVEL,
     & RELHUM,TEMP)
C
C     THIS ROUTINE RETURNS METEOROLOGICAL DATA FOR A GIVEN POINT.
C     VER 2.00
C
C     PROGRAMMER - JOHN SUNDERSON,JR.
C
C         INPUTS
C     X       - X COORDINATE OF GIVEN POINT
C     Y       - Y COORDINATE OF GIVEN POINT
C     TIMEI   - TIME OF METEOROLOGICAL MEASUREMENT DATA TO BE RETURNED
C               (SECONDS FROM BEGINNING OF DAY)
C     IDAY    - DAY OF YEAR FOR OBSERVATION
C         INPUTS FROM COMMON /MET3/
C     WANS    - DATA TYPE IN USE
C     T850C    - TEMPERATURE AT 850 MB IN DEGREES C
C     DP850C  - DEW POINT AT 850 MB IN DEGREES C
C     WSMPM8  - WIND SPEED IN METERS/MIN AT 850 MB
C     WDR8    - WIND DIRECTION IN RADIANS FROM NORTH
C               (DIRECTION WIND MOVES)
C     KUADW   - QUADRANT NO. FOR WIND STORED (NOT USED)
C     XOW     - MINIMUM X COORDINATE OF WIND FIELD
C     YOW     - MINMUM Y COORDINATE OF WIND FIELD
C     XMW     - MAXIMUM X COORDINATE OF WIND FIELD
C     YMW     - MAXIMUM Y COORDINATE OF WIND FIELD
C     DXWI    - X INCREMENT FOR WIND GRID
C     DXWII   - INVERSE OF DXWI
C     DYWI    - Y INCREMENT FOR WIND GRID
C     DYWII   - INVERSE OF DYWI
C     DXW02   - DXWI/2
C     DYW02   - DYWI/2
C     NXWP    - NUMBER OF POINTS IN X DIRECTION
C     NYWP    - NUMBER OF POINTS IN Y DIRECTION
C     NPTSW   - TOTAL NUMBER OF POINTS
C     WINDSG  - WIND SPEED GRID (METERS/MIN)
C     WINDDG  - WIND DIRECTION GRID RADIANS)
C         OUTPUTS
C     WAS     - WIND AZIMUTH FROM TRUE NORTH
C     WVEL    - WIND VELOCITY
C     SDWAS   - STANDARD DEVIATION OF WAS
C     SDWVEL  - STANDARD DEVIATION OF WVEL
C     RELHUM  - RELATIVE HUMIDITY
C     TEMP    - TEMPERATURE (DEGREES C)
C
      COMMON/MET3/WANS,T850C,DP850C,WSMPM8,WDR8,
     &            KUADW,XOW,YOW,XMW,YMW,DXWI,DXWII,DYWI,DYWII,
     &            DXW02,DYW02,NXWP,NYWP,NPTSW,WINDSG(7,7),WINDDG(7,7),
     &            DVSURR,WVSURM,RHSUR,TSURC
      COMMON/SPLMET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
      COMMON /WARNS/ IWARN(3)
      REAL MAG1,MAG2,MAG3,MAG4
      INTEGER WANS,GEOS,SURF
C
      DATA GEOS/4HGEOS/,SURF/4HSURF/
C
C     ************************
C
```

```
C
C     WHICH REGIEME
C
      IF(WANS.EQ.SURF)GO TO 10
C
C     GEOSTROPHIC CASE
C
      CALL SWIND(WSMPM8,WDR8,TDOM,TIMEI,HEIGHT(X,Y),1.0,
     & WVEL,WAS)
      DEWPT=FDEWPT(X,Y,TIMEI,DP850C,WVEL,WAS)
      TEMP=FTEMP(X,Y,TIMEI,T850C,IDAY)
      RELHUM=FRHUM(DEWPT,TEMP)
      SDWAS=0.0
      SDWVEL=0.0
      GO TO 14
C
C     SURFACE MET DATA
C
   10 WAS=0.0
      WVEL=0.0
      SDWAS=DVSURR
      SDWVEL=WVSURM
      RELHUM=RHSUR
      TEMP=TSURC
      IF(X0W.LE.X .AND. X.LE.XMW
     &                .AND.
     &    Y0W.LE.Y .AND. Y.LE.YMW)GO TO 15
      XOUT=X*3.28084
      YOUT=Y*3.28084
      ITMT=IFIX(TIMEI+0.000001)
      ITH=ITMT/60
      ITM=ITMT-ITH*60
      IT1=ITH/10
      IT2=ITH-IT1*10
      IT3=ITM/10
      IT4=ITM-IT3*10
      IF(IWARN(3).EQ.0)PRINT    2001,IT1,IT2,IT3,IT4
      IWARN(3)=1
C
C     LOCATE LOWER LEFT MET DATA
C
   15 IX=IFIX((X-X0W)*DXWII)+1
      IX=MIN0(IX,NXWP-1)
      IX=MAX0(IX,1)
      IY=IFIX((Y-Y0W)*DYWII)+1
      IY=MAX0(1,MIN0(IY,NYWP-1))
```

```
C
C     FIND MAGNITUDE AND DIRECTION OF WIND AT SURROUNDING POINTS
C
      MAG1=WINDSG(IX,IY)
      MAG2=WINDSG(IX,IY+1)
      MAG3=WINDSG(IX+1,IY+1)
      MAG4=WINDSG(IX+1,IY)
      DIR1=WINDDG(IX,IY)
      DIR2=WINDDG(IX,IY+1)
      DIR3=WINDDG(IX+1,IY+1)
      DIR4=WINDDG(IX+1,IY)
C
C     PREFORM DOUBLE LINEAR INTERPOLATION
C
      X1=FLOAT(IX-1)*DXWI+XOW
      Y1=FLOAT(IY-1)*DYWI+YOW
      DELX=AMAX1(0.0,AMIN1(DXWI,X-X1))
      DELY=AMAX1(0.0,AMIN1(DYWI,Y-Y1))
      FRAC=DELY*DYWII
      WVEL1=(MAG2-MAG1)*FRAC+MAG1
      WVEL2=(MAG3-MAG4)*FRAC+MAG4
      WVEL=WVEL1+(WVEL2-WVEL1)*DELX*DXWII
      WAS1=(DIR2-DIR1)*FRAC+DIR1
      WAS2=(DIR3-DIR4)*FRAC+DIR4
      WAS=WAS1+(WAS2-WAS1)*DELX*DXWII
C
   14 RETURN
 2001 FORMAT(3X,45H***WARNING*** DATA BASE EXCEEDED AT TIME = ,
     & 2I1,1H:,2I1/17X,36HALL DATA AFTER THIS TIME ARE SUSPECT)
      END
```

34.0 FDEWPT Version 1.00 25 June 1975

1. Purpose

This routine computes the dewpoint temperature at a given location and time of day.

2. Arguments

INPUTS

X,Y	-	the coordinates of the location
TOD	-	the time of day (minutes from midnight)
DP850	-	the 850 millibar dewpoint temperature (°C)
WVEL	-	the general wind speed (meter/min)
WAS	-	the general wind direction (azimuth in radians)

OUTPUTS

FDEWPT	-	the dewpoint (°C)

3. Computation Procedure

Functions that model the dewpoint temperature have been developed for the south slope of the San Bernardino mountains.[3] In the absence of more general models, these functions may be used to generate the meteorological data required by the fuel moisture content models.

The input arguments are converted into local units. The time of day (t) is in hours and the wind direction (α) is in degrees. The 850 millibar dewpoint (T_{850}) however, is in degrees centigrade. The elevation (E) for the input location is retrieved via the subroutine HEIGHT and converted into feet.

Two wind conditions are considered in the model.[3] If the general wind is greater than 14 mph and from the northeast, Santa Ana conditions are assumed to prevail; otherwise marine air is taken to be present. In the marine air model, the height of the base of the transition layer, h_t, between the marine air and the dry air above it is calculated from

$$h_t = 2868 + 1281 \sin(0.262t + 3.503) + 332 \sin(0.524t + 5.888). \tag{1}$$

The dew point temperature at 1,000 feet, T_{1000}, is computed from

$$T_{1000} = 1.84T_{850} + 50.13. \tag{2}$$

Three separate elevation dependent cases are then considered; these are briefly described below:

<u>CASE I</u> -- $E > h_t + 800$ (3)

The elevation is above the transition layer, and the dewpoint is calculated by

$$T_{dp} = 0.0025(h_t + 800-E) + T_{1000} - 10.0. \tag{4}$$

<u>CASE II</u> -- $h_t < E \leq h_t + 800$

The elevation is in the transition layer, and the dewpoint formula is given by

$$T_{dp} = 0.00625(h_t - E) + T_{1000} - 5.0. \tag{5}$$

<u>CASE II</u> -- $E \leq h_t$

The elevation is below the transition layer, and the dewpoint is computed from

$$T_{dp} = T_{1000} + 5(1000-E)/(h_t-1000). \quad (6)$$

In the Santa Ana wind condition model, the dewpoint is correlated with the 850 millibar dewpoint, and temperature as follows;

$$T_{dp} = 9.42 - 4.15 \times 10^{-4} + 1.227T_{850} + \Delta T_{dp}, \quad (7)$$

where ΔT_{dp} is given by

$$\Delta T_{dp} = -2.324 + 2.926 \sin(.262t + 4.254) + 0.527 \sin(0.524t + 4.247). \quad (8)$$

The calculated dewpoint is then converted to degrees centigrade for output to the calling program.

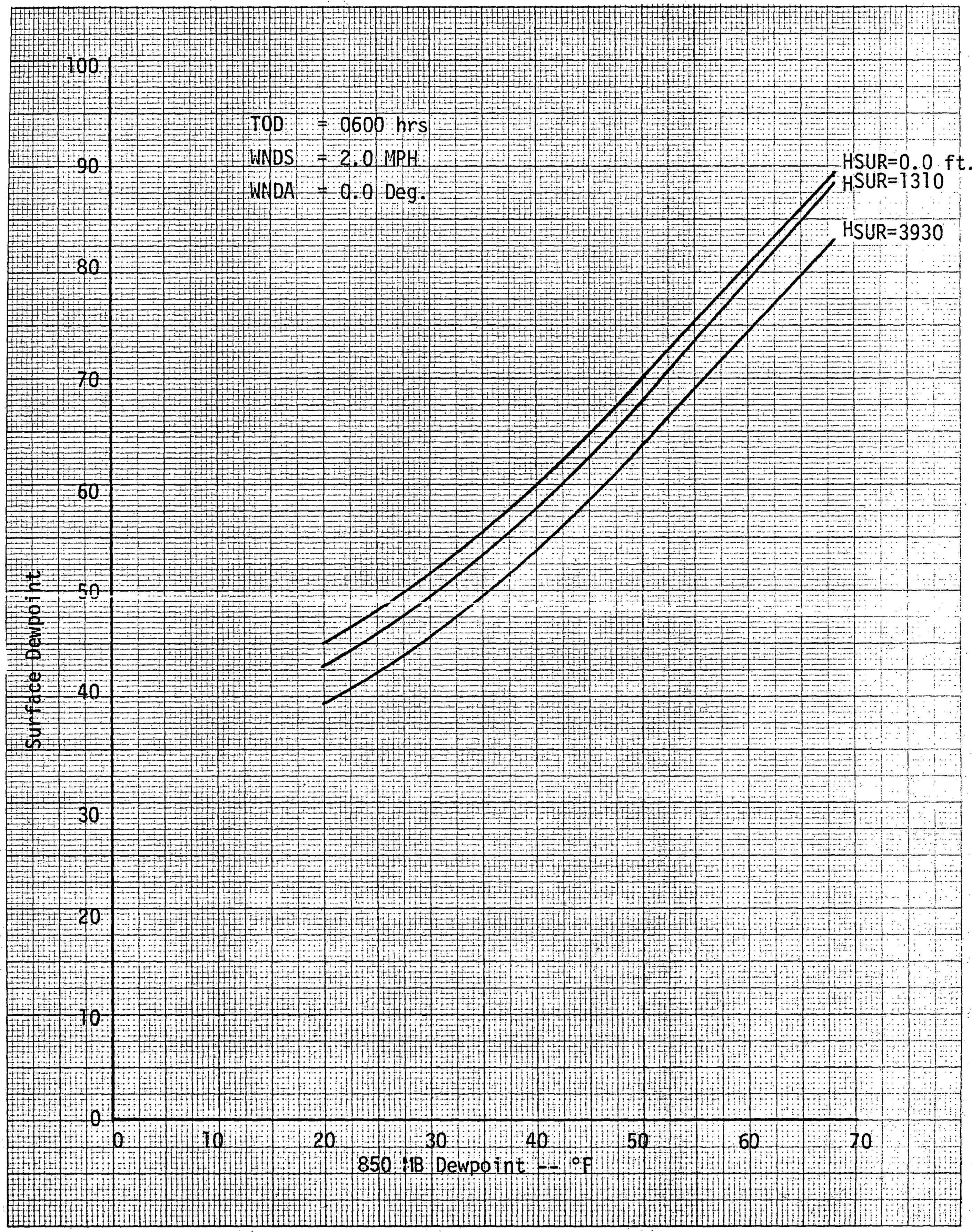
4. MODEL RESULTS

Surface dewpoint is shown in Figure 47 plotted against 850MB dewpoint for three values of surface height.

5. Flow Chart

Not required.

Figure 47 Surface Dewpoint as a Function of 850 MB Dewpoint



```
      FUNCTION FDEWPT(X,Y,TOD,DP850,WVEL,WAS)
C
C     THIS FUNCTION COMPUTES THE DEWPOINT TEMPERATURE AT A POINT
C     VERSION 1.00
C
C     PROGRAMMER - J KEEFER
C     TECHNOLOGY - MCCUTCHAN,RYAN,SCHROEDER
C
C         INPUTS
C     X,Y     - COORDINATES OF THE POINT
C     TOD     - TIME OF DAY IN MINUTES FROM MIDNIGHT
C     DP850   - THE 850 MILLIBAR DEW POINT IN DEGREES CENTIGRADE
C     WVEL    - THE GENERAL WIND SPEED(METER/MINUTES)
C     WAS     - THE GENERAL WIND DIRECTION(AZIMUTH IN RADIANS)
C         OUTPUT
C     FDEWPT - THE DEW POINT(DEG. CENTIGRADE AT THE POINT X,Y FOR TOD)
C
C         ROUTINES USED
C     HEIGHT
C
      COMMON /CNSINT/ RE,PI,PI0180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
C
C     CONVERT INPUT VARIABLES TO THE UNITS USED BY THE FORMULAS
C         TIME IN HOURS, ANGLE IN DEGREES, ELEVATION IN FEET
C
      TIME = TOD/60.0
      IF(WAS.LT.0.0) WAS = WAS + TWOPI
      IF(WAS.GT.TWOPI) WAS = WAS - TWOPI
      ASW = WAS/PI0180
      ELEV = HEIGHT(X,Y)*3.281
C
C     DETERMINE WIND CONDITION
C
      IF(WVEL.GE.475.5.AND.ASW.GT.180.0
     &                 .AND.ASW.LT.270.0)  GO TO 20
C
C     MARINE WIND CONDITION DEWPOINT(DEGREES FAHRENHEIT)
C
      HTL = 2.868E3 + 1.281E+3*SIN(0.262*TIME + 3.504) +
     &                     3.32E2*SIN(0.524*TIME + 5.888)
      DP1000 = 1.84*DP850 + 50.13
      HTL8= HTL + 800.0
      IF(ELEV.GT.HTL8) DEWPT = 2.5E-3*(HTL8-ELEV) + DP1000 - 10.0
      IF(ELEV.LE.HTL8.AND.ELEV.GT.HTL) DEWPT = 6.25E-3*(HTL-ELEV) +
     &                                          DP1000 - 5.0
      IF(ELEV.LE.HTL) DEWPT = 5.0*((1000.0-ELEV)/(HTL-1000.0)) + DP1000
      GO TO 50
C
C     SANTA ANA WIND CONDITIONS DEWPOINT(DEGREES FAHRENHEIT)
C
   20 CONTINUE
      DELDP = -2.324 + 2.926*SIN(.262*TIME + 4.254) +
     &           0.527*SIN(0.524*TIME + 4.347)
      DEWPT = 9.42 - 4.15E-4*ELEV + 1.227*DP850 + DELDP
C
C     CONVERT DEWPOINT TO DEGREES CENTIGRADE
C
   50 CONTINUE
      FDEWPT= 0.556*(DEWPT - 32.)
C
      RETURN
      END
```

1. Purpose

This routine determines the topographic characteristics at a specified point.

2. Arguments

Input

KOUT	=	Output option = 1 SLOPE and SLOPN are calculated = 2 SLOPE, SLOPN, and ASPCT are calculated = 3 SLOPE, SLOPN, ASPCT and SLP are calculated, BRNG is required as input for this option only.
XIN	=	X coordinate of given point
YIN	=	Y coordinate of given point
BRNG	=	Bearing for which slope (SLP) is to be calculated for KOUT=3

Output

ZOUT	=	Z coordinate of given point
SLOPE	=	East slope of given point
SLOPN	=	North slope of given point
ASPCT	=	Aspect of given point
SLP	=	Slope in direction given by BRNG.

3. Procedure

Altitude - double linear interpolation (see HEIGHT).

East and North Slope - 4 points surrounding the given point (one each in North South, East and West directions) are chosen, slopes computed from these.

Aspect - $\pi - \tan^{-1}$ (East slope/North slope). For the case where aspect is non-existant, ASPCT is set to -2π.

SLP - The East and North slopes are combined using the sine and cosine of the bearing, respectively.

4. Comments

These algorithms were gathered from a preliminary test routine, and are based on the definitions of each quantity. This implementation requires no changes if the topographic data base form changes, since it interfaces through HEIGHT.

5. Flow Chart

See Figure 48.

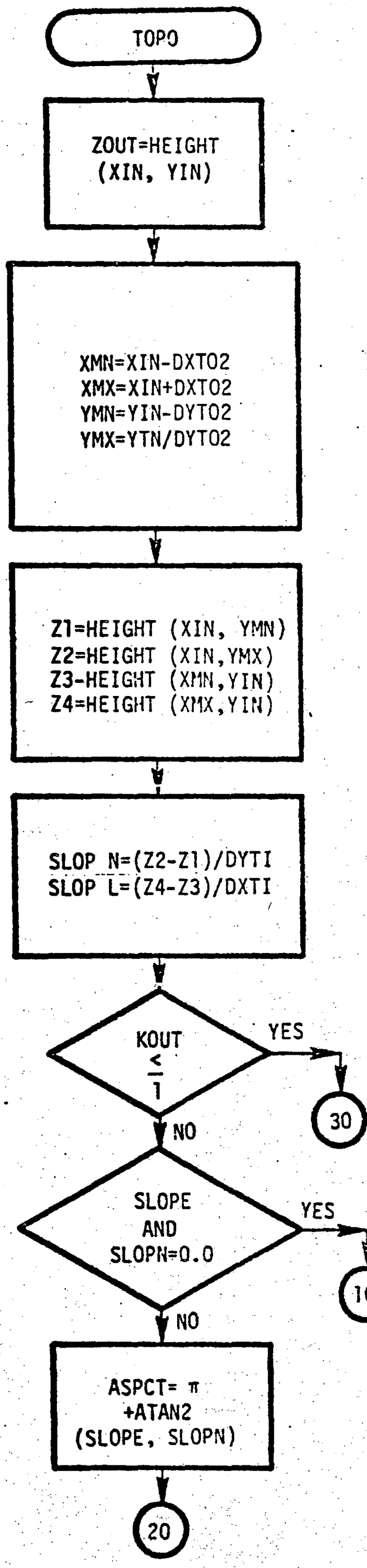


Figure 48 Logic Structure for Routine TOPO

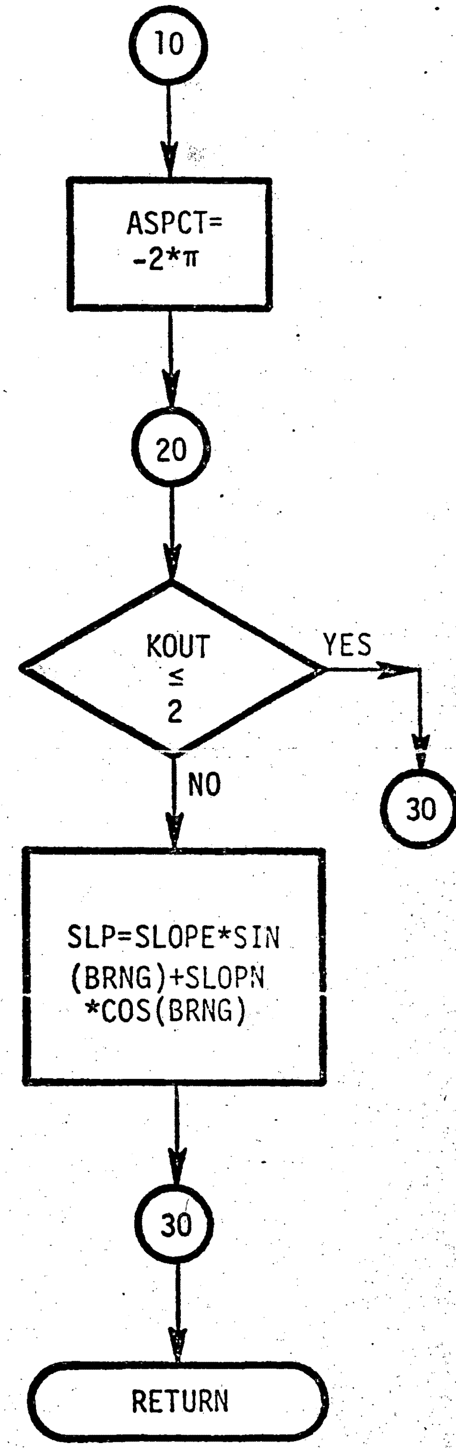


Figure 48 Logic Structure for Routine TOPO (Cont'd)

```
      SUBROUTINE TOPO(KOUT,XIN,YIN,BRNG,
     &                 ZOUT,SRUFO,SLOPE,SLOPN,ASPCT,SLP)
C
C     THIS ROUTINE DETERMINES TOPOGRAPHIC CHARACTERISTICS OF A POINT
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     KOUT   - OUTPUT OPTION
C              =1 SLOPE AND SLOPN ARE CALCULATED
C              =2 SLOPE, SLOPN,AND ASPCT ARE CALCULATED
C              =3 SLOPE, SLOPN,ASPCT, AND SLP ARE CALCULATED
C              =4 SLOPE,SLOPN,ASPCT, AND SLP IN DIRECTION OF
C                 ASPCT ARE CALCULATED
C                 BRNG REQUIRED AS INPUT FOR THIS OPTION ONLY
C     XIN    - COORDINATE OF GIVEN POINT
C     YIN    - Y COORDINATE OF GIVEN POINT
C     BRNG   - BEARING FOR WHICH SLOPE (SLP) IS TO BE CALCULATED
C              KOUT = 3 ONLY (RADIANS)
C         OUTPUTS
C     ZOUT   - Z COORDINATE OF GIVEN POINT
C     SLOPE  - EAST SLOPE AT (XIN,YIN)   (RADIANS)
C     SLOPN  - NORTH SLOPE AT (XIN,YIN)   (RADIANS)
C     ASPCT  - ASPECT AT (XIN,YIN)
C     SLP    - SLOPE IN DIRECTION GIVEN BY BRNG (RADIANS)
C              FOR OPTION 4 ,  SLOPE IN DIRECTION GIVEN BY ASPCT
C
C         ROUTINES USED
C     HEIGHT,SIN,COS
C
      COMMON /CNSTNT/ RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
      COMMON/TOPO2/KUADT,XOT,YOT,XMT,YMT,DXTI,DXTII,DYTI,DYTII,SRUF,
     &              DXTO2,DYTO2,NXTP,NYTP,NPTST,HBC(51,51)
C
C ****************************
C
      SRUFO=SRUF
C     FIND ALTITUDE OF GIVEN POINT AND LOCATION AND ALTITUDE
C     OF  FOUR SURROUNDING POINTS
C
      ZOUT=HEIGHT(XIN,YIN)
      XMN=XIN-DXTO2
      XMX=XIN+DXTO2
      YMN=YIN-DYTO2
      YMX=YIN+DYTO2
      Z1=HEIGHT(XIN,YMN)
      Z2=HEIGHT(XIN,YMX)
      Z3=HEIGHT(XMN,YIN)
      Z4=HEIGHT(XMX,YIN)
```

```
C
C     CALCULATE EAST AND NORTH SLOPES
C
      SLOPE=(Z4-Z3)*DXTII
      SLOPN=(Z2-Z1)*DYTII
C         SELECT OPTION
      IF(KOUT.LE.1)GO TO 30
C
C     CHECK FOR FLAT GROUND BEFORE CALCULATING ASPECT
C
      IF(SLOPE.EQ.0.0.AND.SLOPN.EQ.0.0)GO TO 10
      ASPCT=PI+ATAN2(SLOPE,SLOPN)
      GO TO 20
   10 ASPCT=-TWOPI
C         SELECT OPTION
   20 IF(KOUT.LE.2)GO TO 30
C
C     CALCULATE SLOPE IN GIVEN DIRECTION
C
      BRN=BRNG
      IF(KOUT.EQ.4)BRN=ASPCT
      SLP=SLOPE*SIN(BRN)+SLOPN*COS(BRN)
C
   30 CONTINUE
      RETURN
      END
```

36.0 FRHUM Version 1.00 10 May 1976

1. PURPOSE

This function computes the relative humidity for a given temperature and dewpoint.

2. ARGUMENTS

INPUTS

DEWPT - the dewpoint (°C)

TEMP - the temperature (°C)

OUTPUTS

FRHUM - the relative humidity in percent

3. PROCEDURE

The input temperatures are converted to degrees Fahrenheit. The dependence of relative humidity (h_r) on dewpoint (T_{dp}) and temperature (T) is given by:[3]

$$h_r = \max [100 \exp(7482.6 (T_p - D_p)) . 100.0] \qquad (1)$$

where T_p = temperature (°C) and D_p = dewpoint (°C)

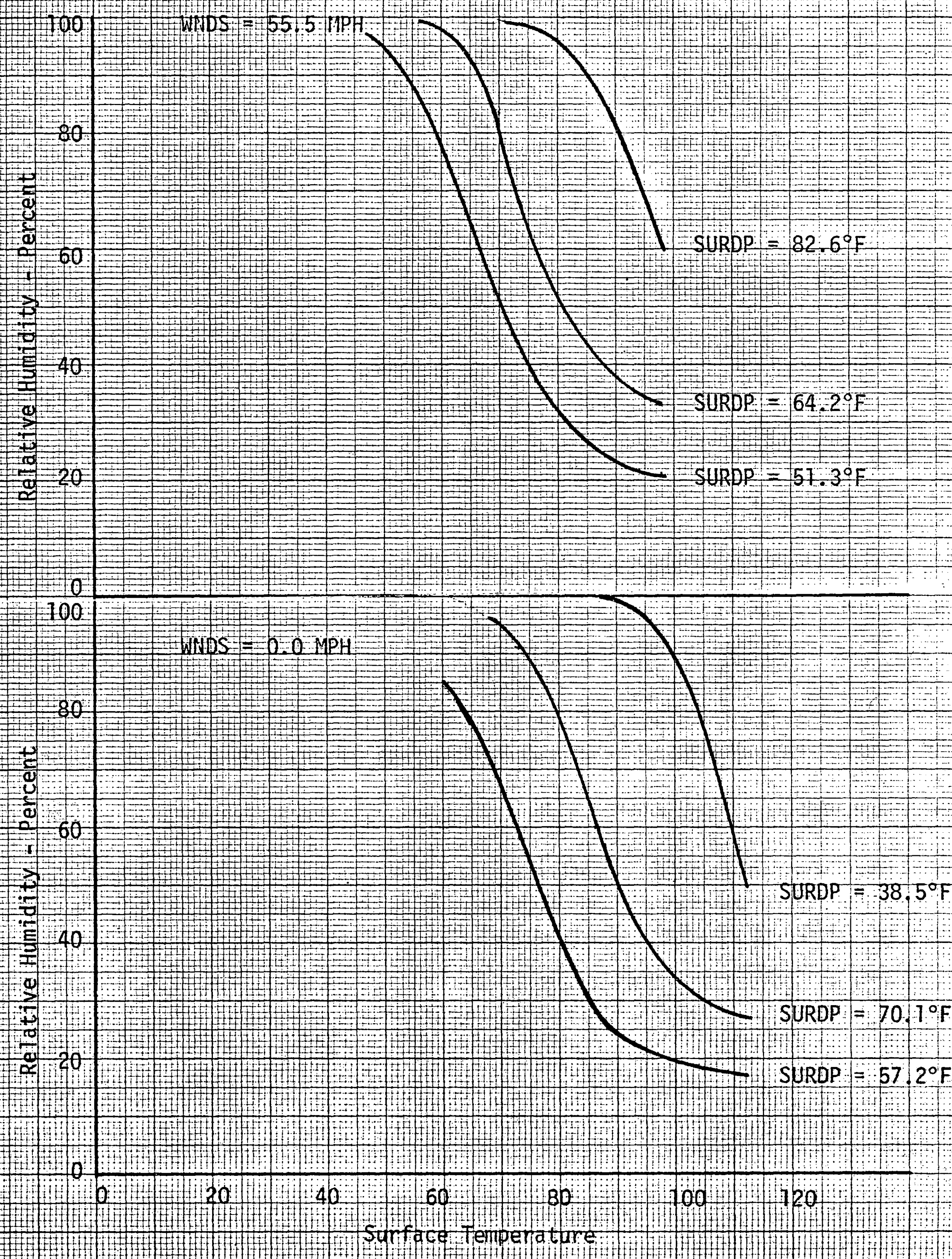
4. MODEL RESULTS

Relative humidity is shown in Figure 49 plotted against surface temperature with surface dewpoint as a parameter, for two values of surface wind speed.

5. FLOW CHART

Not required.

Figure 49. Relative Humidity Versus Surface Temperature at Two Wind Speeds



1. PURPOSE

This routine calculates local solar insolation considering surface orientation and sun position.

2. ARGUMENTS

INPUT

SLP	=	slope (rad relative to local horizontal)
ASP	=	azimuth angle of surface normal
ASUN	=	polar angle of the sun (rad from vertical)
GSUN	=	aximuth angle of sun

OUTPUT

PHI	=	insolation function

3. PROCEDURE

The quantities ϕ_n = SLP, θ_n = ASP and ϕ_s = ASUN, θ_s = GSUN express the direction of the surface normal ($\hat{n}$) and the sun $\hat{a}_s$, respectively in polar ENV* coordinates. Expressed in rectangular ENV coordinates these are

$$X_n = \sin\phi_n \cos\theta_r \qquad X_s = \sin\phi_s \cos\theta_s$$
$$Y_n = \sin\phi_n \sin\theta_n \qquad Y_s = \sin\phi_s \sin\theta_s$$
$$Z_n = \cos\phi_n \qquad Z_s = \cos\phi_x$$

The insolation function Φ is defined by the scalar product

$$\Phi = \hat{a}_s \cdot \hat{n} = X_n X_s + Y_n Y_s + Z_n Z_s.$$

*ENV - East North Vertical coordinates

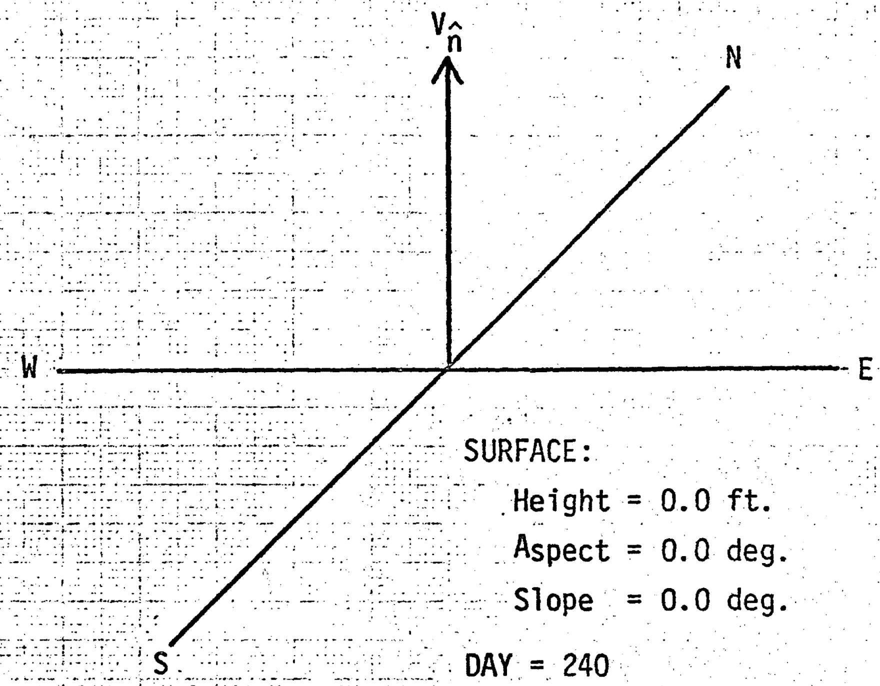
4. MODEL RESULTS

Figures 50 through 53 present the insolation factor Φ plotted against time of day, for one day of the year and four sets of values of: surface height, surface aspect, surface slope.

5. Flow Chart

Not required

Figure 50 Solar Insolation Factor Versus Time of Day



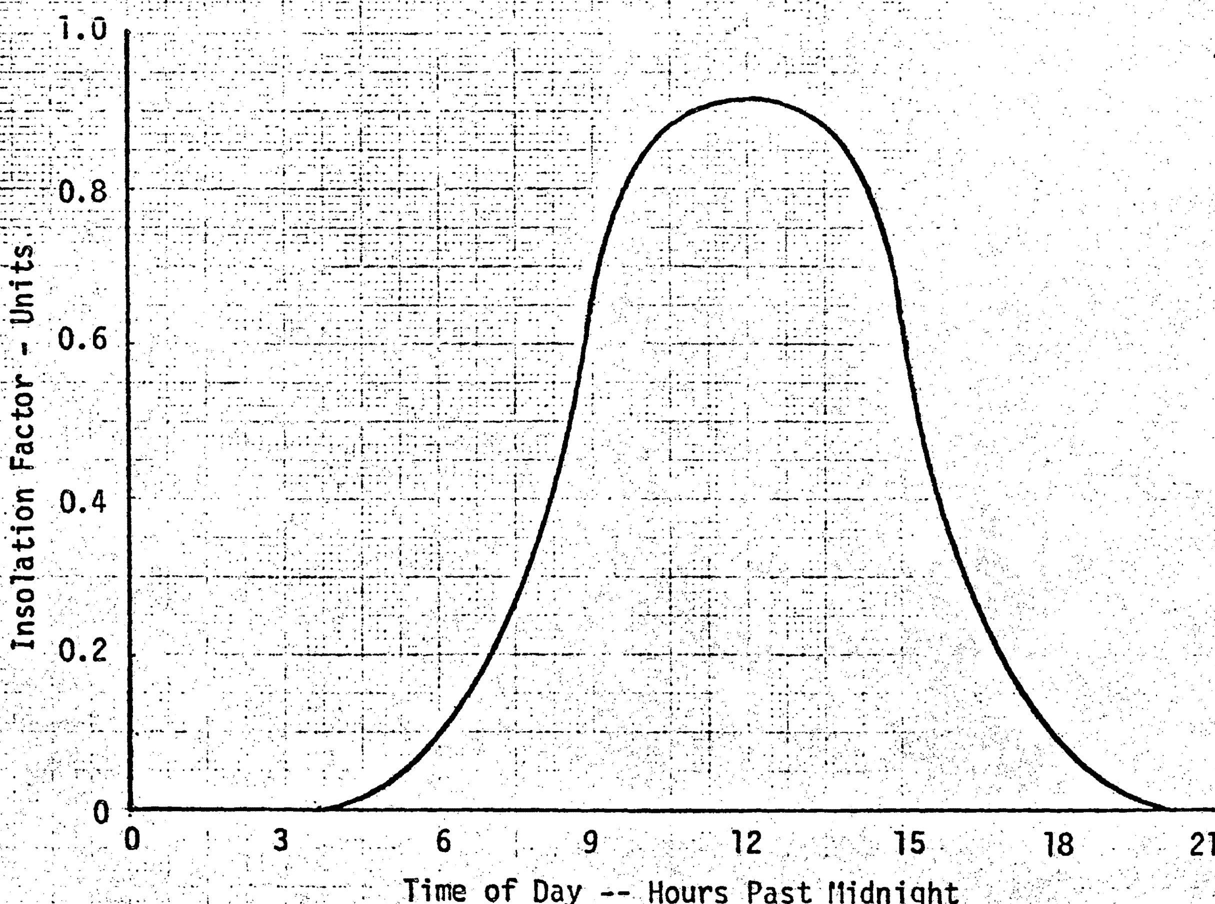


Figure 51 Solar Insolation Factor Versus Time of Day

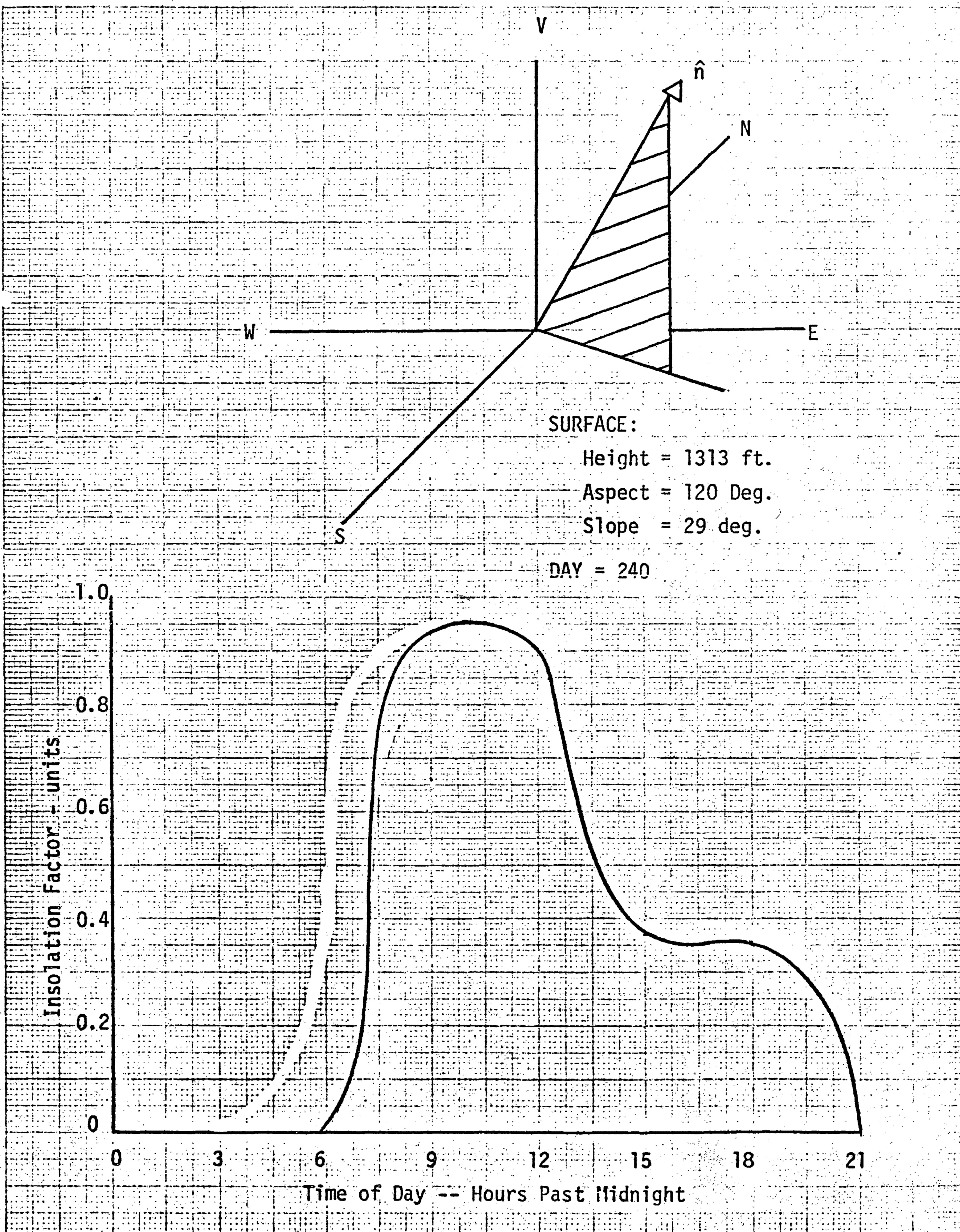


Figure 52 Solar Insolation Factor Versus Time of Day

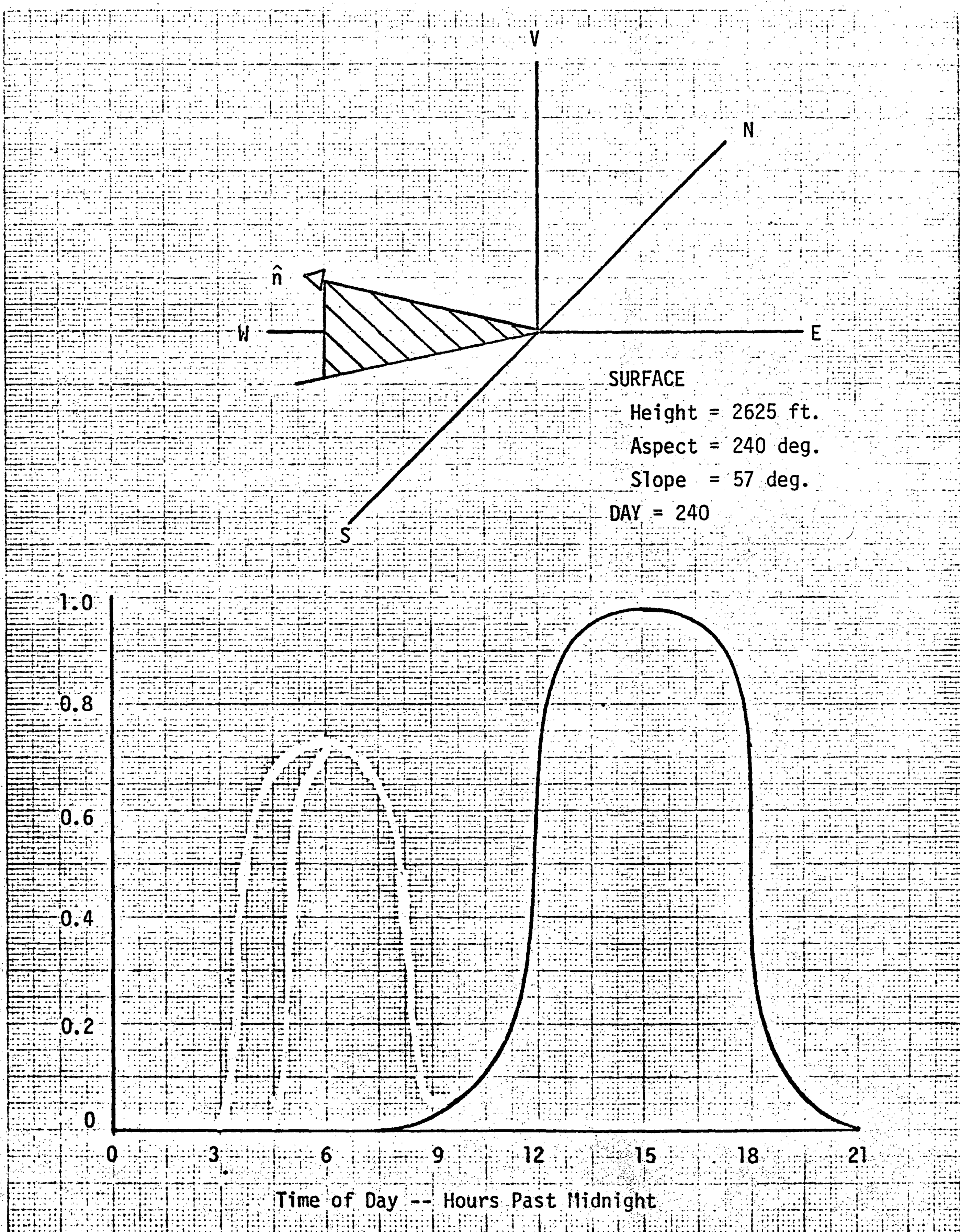


Figure 52   Solar Insolation Factor Versus Time of Day

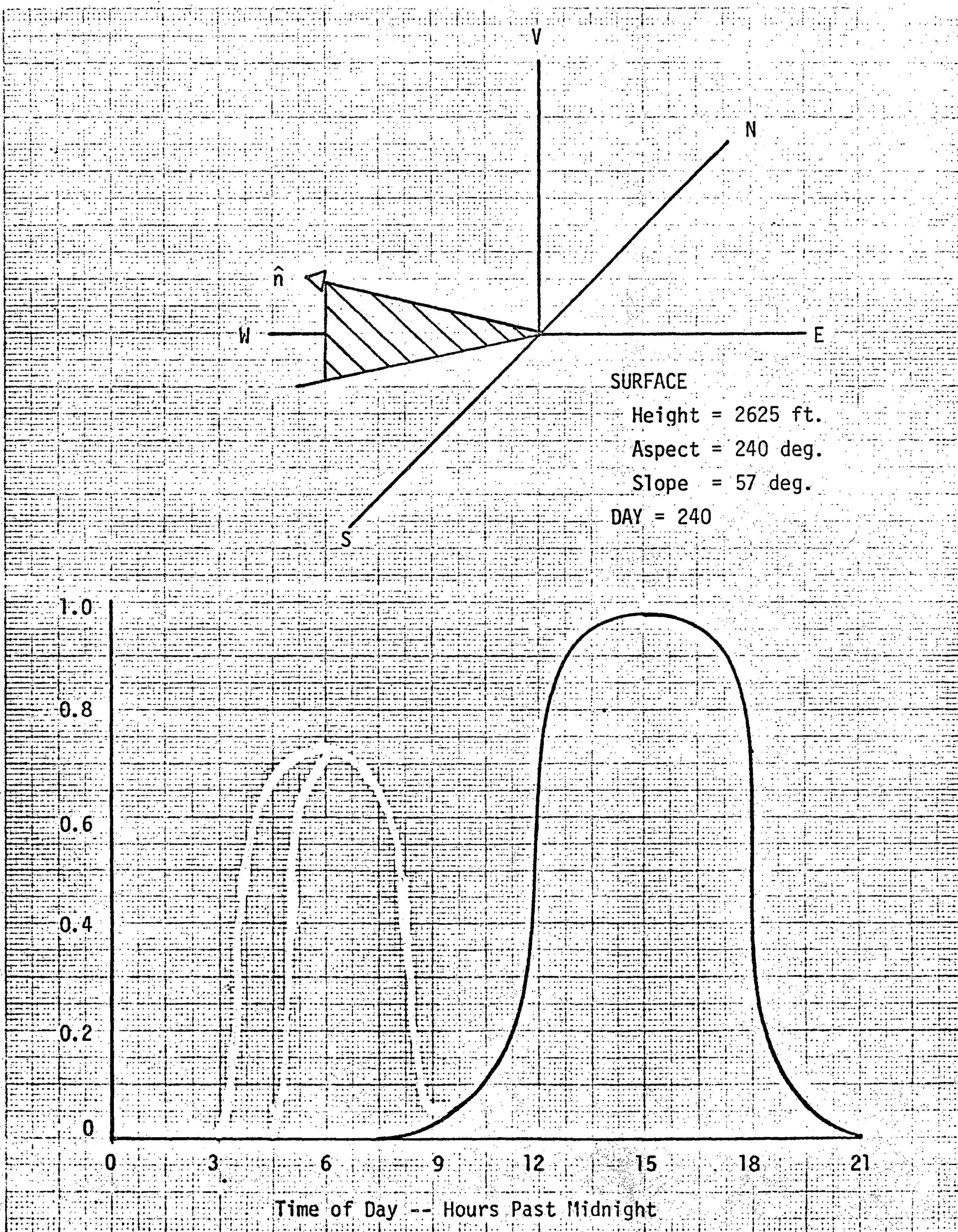
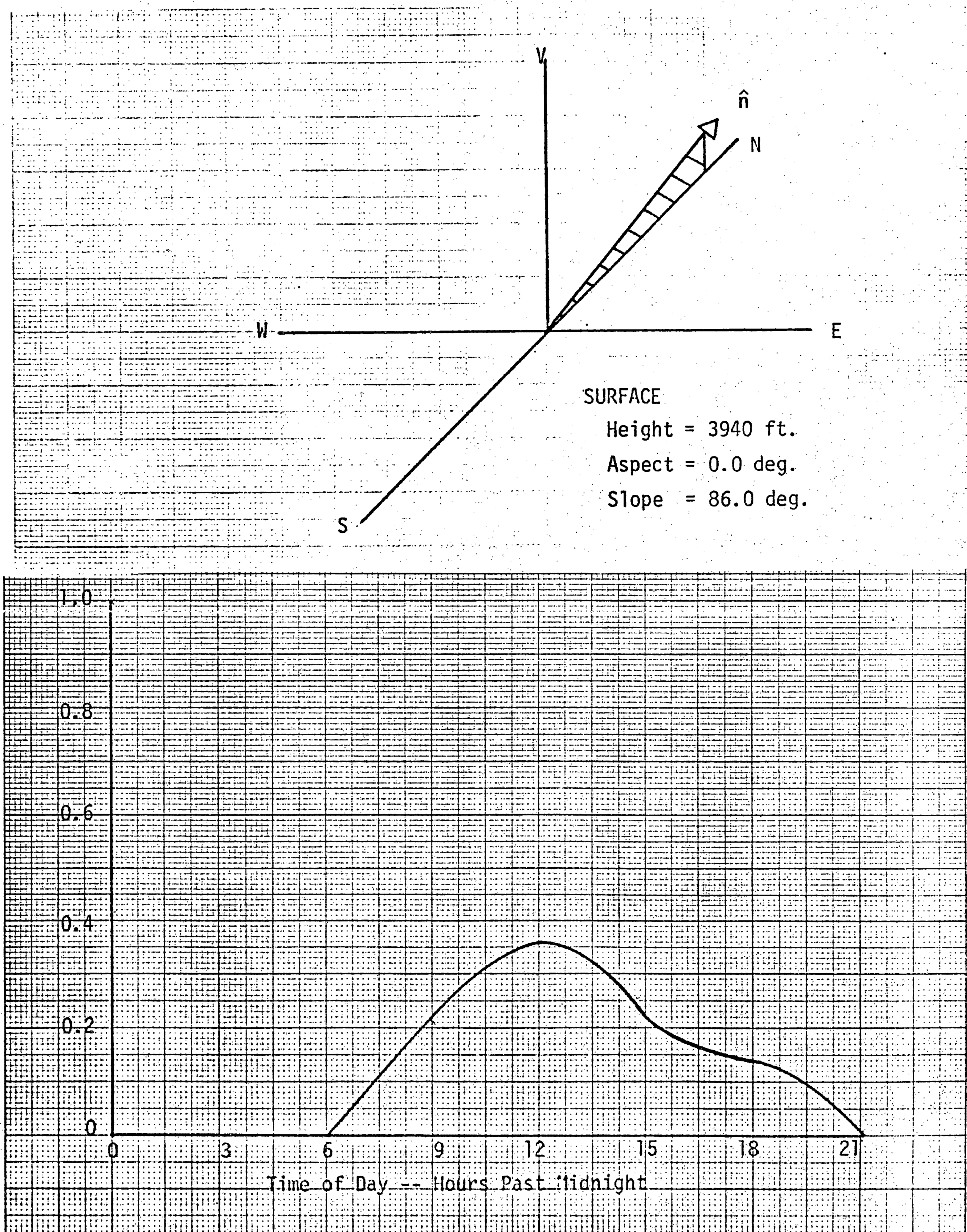


Figure 53 Insolation Factor Versus Time of Day



```
      SUBROUTINE INSOLA(HALFPI,SLP,ASP,ASUN,GSUN,COSPHI)
C
C     THIS ROUTINE CALCULATES SURFACE POLAR ANGLES IN E, N, V
C     INSOLATION AT SURFACE
C     SURFACE POLAR ANGLES
C     VER 1.00  04/21/76
C
C     PROGRAMMER - DR. JAMES SANDERLIN
C
C         INPUTS
C     HALFPI - 3.14/2
C     SLP    - SLOPE AT POINT OF INTEREST
C     ASP    - ASPECT AT POINT OF INTEREST
C     ASUN   - POLAR ANGLE OF SUN
C     GSUN   - AZIMUTHAL ANGLE OF SUN
C
C         OUTPUT
C     COSPHI - COSINE DEPENDENT SOLAR INSOLATION AT SURFACE
C
      PG=SLP
      SINPG=SIN(PG)
      COSPG=COS(PG)
      SINAG=SIN(ASP)
      COSAG=COS(ASP)
C         CARTESIAN COORDINATES
      X1=SINPG*SINAG
      Y1=SINPG*COSAG
      Z1=COSPG
C         RADIATION
      SINPS=SIN(ASUN)
      COSPS=COS(ASUN)
      SINAS=SIN(GSUN)
      COSAS=COS(GSUN)
C         CARTESIAN COORDINATES
      X2=SINPS*SINAS
      Y2=SINPS*COSAS
      Z2=COSPS
C         INSOLATION AS A FUNCTION OF THE ANGLE BETWEEN SOLAR
C         IRRADIATION AND LOCAL SURFACE NORMAL
      COSPHI=ABS(X1*X2+Y1*Y2+Z1*Z2)
      RETURN
      END
```

38.0 SUNANG Version 1.00 9 April 1976

1. PURPOSE

This subroutine calculates local polar angles of the sun for a specified day of the year and time of day.

2. ARGUMENTS

INPUTS

DAY = Julian day of the year

TIMA = local hour angle (rad)

OUTPUT

ASUN = polar angle of sun (rad from vertical)

GSUN = aximuth of sun (rad from north)

3. PROCEDURE

The angle of the sun from local vertical is given by

$$\text{ASUN} = \cos^{-1}(\cos B \cos C + \sin B \sin C \cos \alpha)$$

where

B = 90° - local latitude (∿34.25 degrees for Southern California

C = 90° - δ

δ = doclination of sun

α = local hour angle with respect to the meridian

The local aximuth angle of the sun is given by

$$\text{GSUN} = \gamma = 2 \sin^{-1}(\sin(S-B) \sin(S-A)/(\sin B \sin A))^{1/2}$$

where S = A+B+C for positive values of local hour angle α. For negative values of local hour angle the local azimuth angle of the sun is given by GSUN $= 360° - \gamma$.

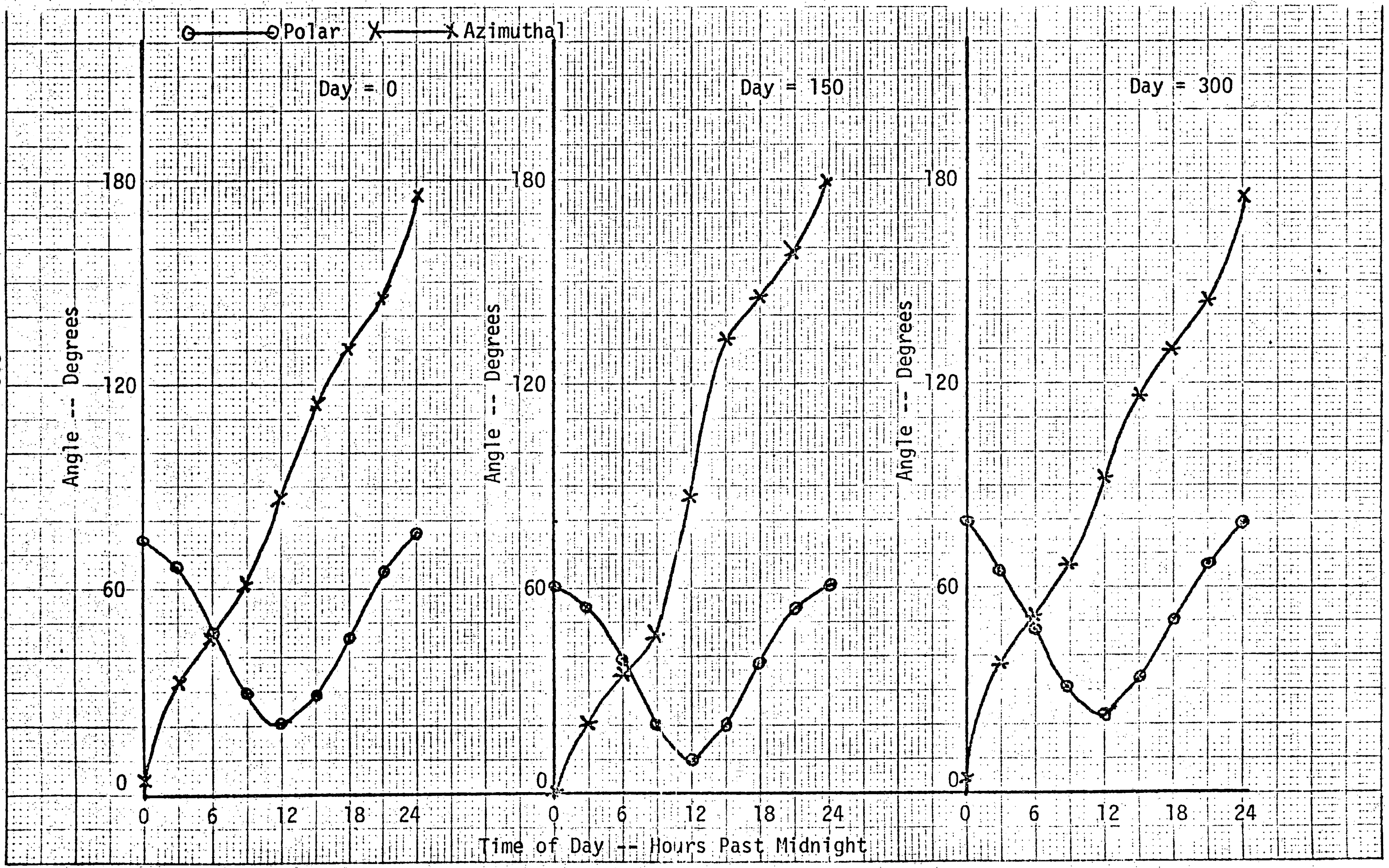
4. MODEL RESULTS

Figure 54 presents three vertical and aximuthal angles of the sun plotted versus time of day for three days of the year and a Southern California location.

5. Flow Chart

Not required.

Figure 54   Polar and Azimuthal Angles of Sun Versus Time of Day with Day of Year as a Parameter



```
      SUBROUTINE SUNANG(PI0180,DAY,TIMA,ASUN,GSUN)
C
C     FUNCTION TO CALCULATE LOCAL SUN POLAR ANGLES FROM
C      LOCAL VERTICAL AND NORTH
C     VER 1.00 04/21/76
C
C     PROGRAMMER - DR. JAMES SANDERLIN
C
C
C         INPUTS
C     PI0180 - DEGREES TO RADIANS CONVERTION
C     DAY    - DAY OF YEAR
C     TIMA   - LOCAL SUN HOUR ANGLE
C
C         OUTPUTS
C     ASUN   - POLAR ANGLE OF SUN
C     GSUN   - AZIMUTHAL ANGLE OF SUN
C
C
C     DECLINATION OF SUN
C
      DEL=PI0180*23.46*SIN(PI0180*(0.9863*DAY-79.88))
C         POLAR ANGLE
      C=PI0180*90.0-DEL
      B=PI0180*(90.0-34.25)
      COSA=COS(B)*COS(C)+SIN(B)*SIN(C)*COS(TIMA)
      A=ACOS(COSA)
C
C     BEARING ANGLE
C
      S=(A+B+C)/2.0
      SINGO2=SQRT((SIN(S-B)*SIN(S-A))/(SIN(B)*SIN(A)))
      G=2.0*ASIN(SINGO2)
      IF(TIMA .GT. 0.0)G=PI0180*360.0-G
      GSUN=G
      ASUN=A
      RETURN
      END
```

39.0 SURFACE TEMPERATURE (FTEMP) Version 2.00 20 April 1976

1. PURPOSE

This function computes local surface temperature in terms of the 850 MB temperature considering local solar insolation and a mean diurnal variation.

2. ARGUMENTS

<u>INPUTS</u>

X,Y	=	coordinates of point of interest
TOP	=	time of day (minutes from midnight)
T850	=	850 MB temperature (°C)
JDOY	=	Day of year (days since 31 December)

<u>OUTPUTS</u>

FTEMP	=	surface temperature (°C)

3. PROCEDURE

In the standard atmosphere the temperature at height h can be expressed in terms of the 850 MB temperature (T_R) as

$$T(T_R,h) = T_R - 0.00648(h-1650.0).$$

The mean diurnal temperature variation f(t) is defined to be

$$f(t) = 0.91 + 0.13 \sin(t - 15.0).$$

The insolation factor Φ is obtained from subroutine ENSOL. Given the 850 MB temperature, the surface height and orientation, day of year and time of day the surface temperature is defined to be

$$T_S = T(T_R,h)f(t)\Phi.$$

4. MODEL RESULTS

Curves of surface temperature are shown in Figures 55 through 58 plotted against time of day, with 850 MB temperature as a parameter and for four values for the set of variables: surface height, surface aspect, and surface slope. It may be observed that these four curves correspond to the four solar insolation curves presented in INSOLA (Para. 38.0).

5. FLOW CHART

Not required.

Figure 55 Surface Temperature Versus Time of Day with 850 MB Temperature as a Parameter

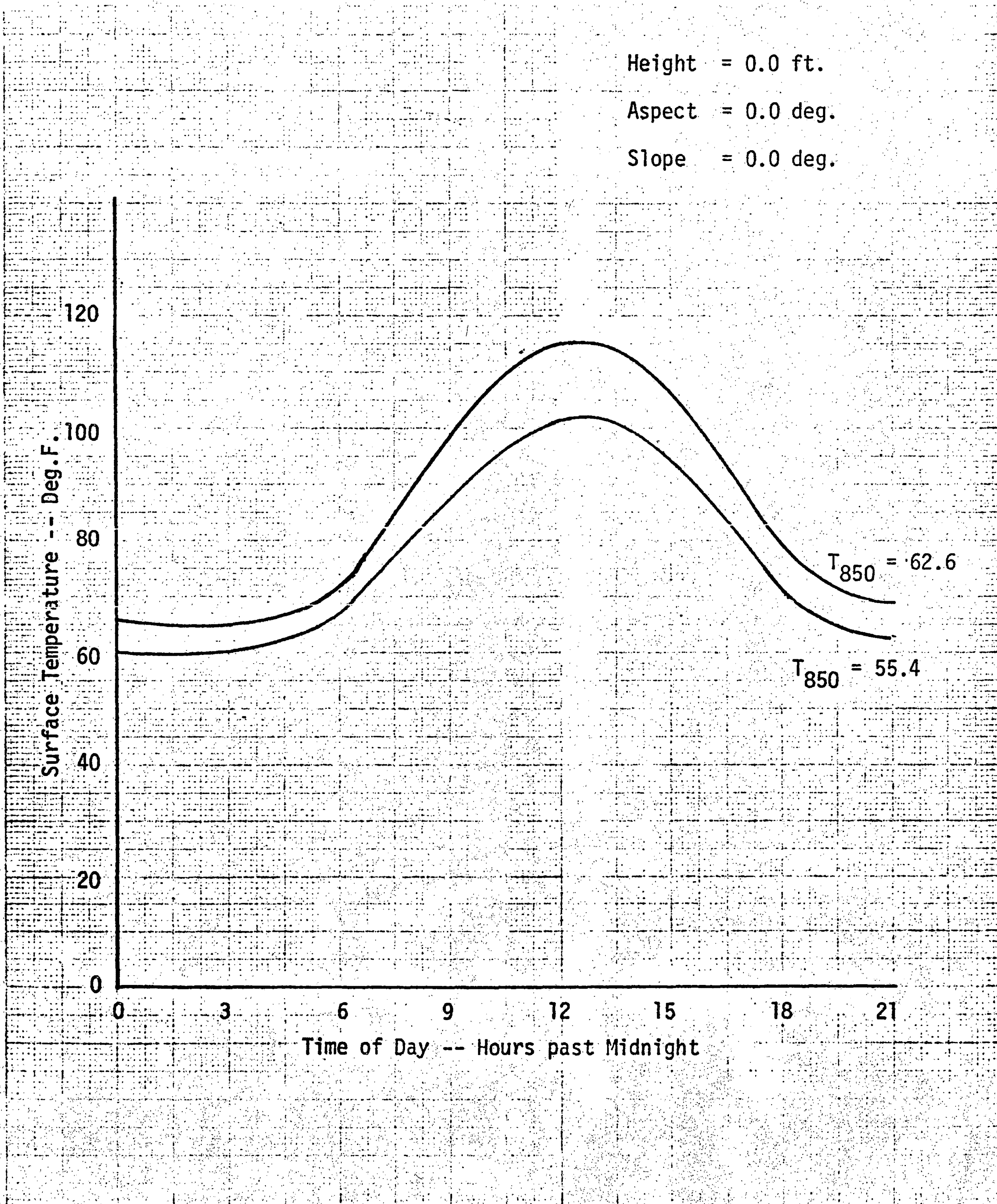


Figure 56 Surface Temperature Versus Time of Day with 850 MB Temperature as a Parameter

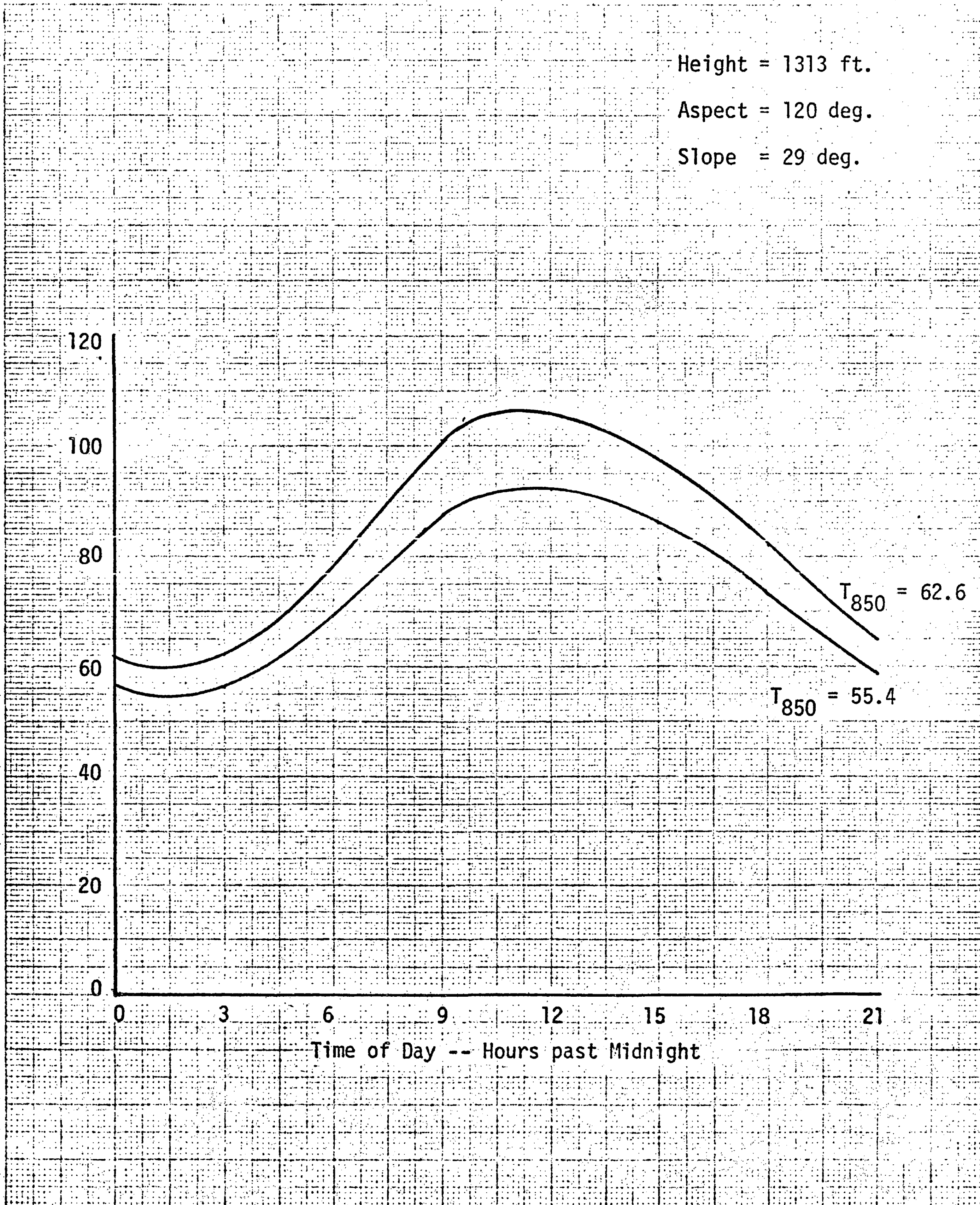


Figure 57 Surface Temperature Versus Time of Day with 850 MB Temperature as a Parameter

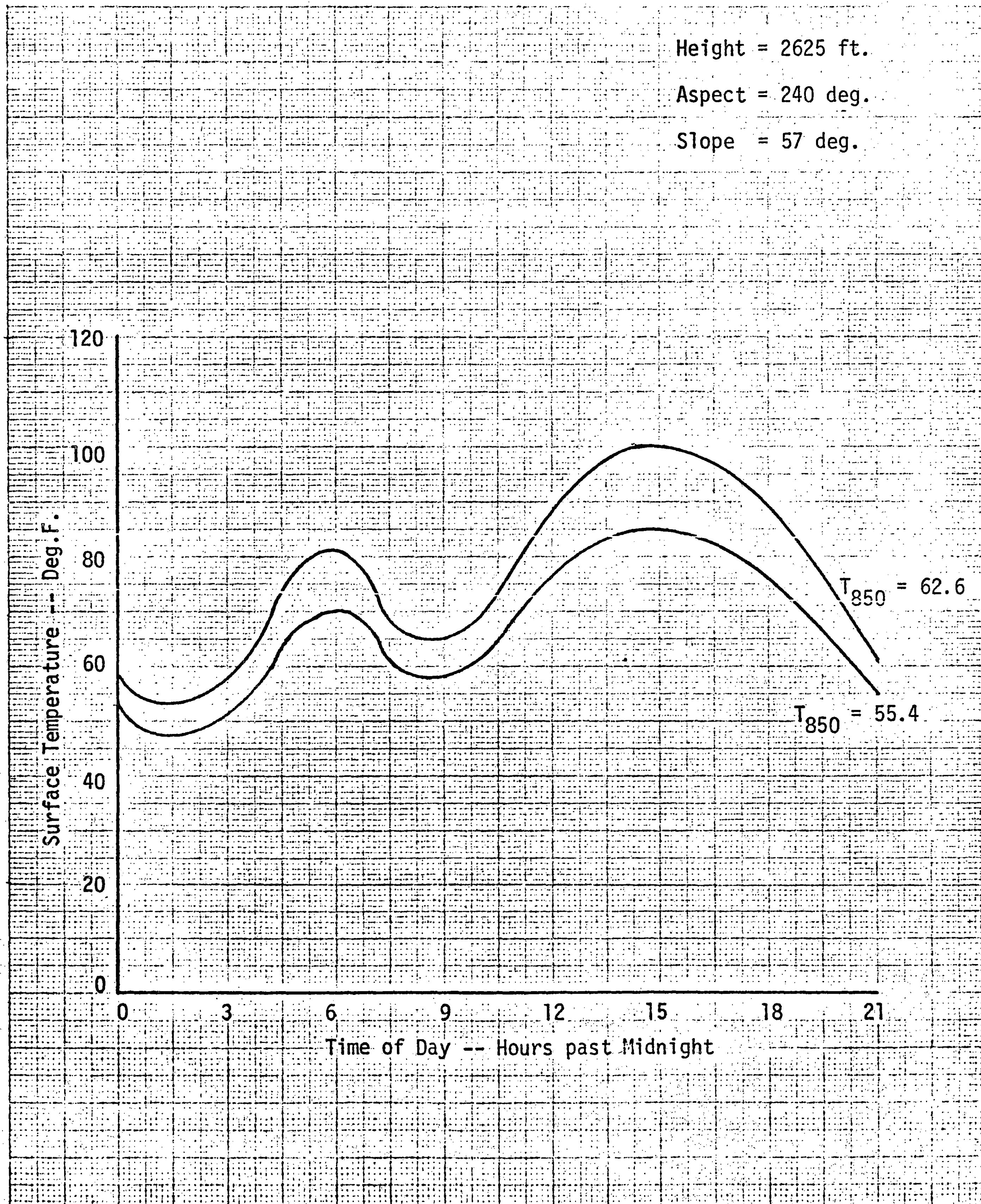
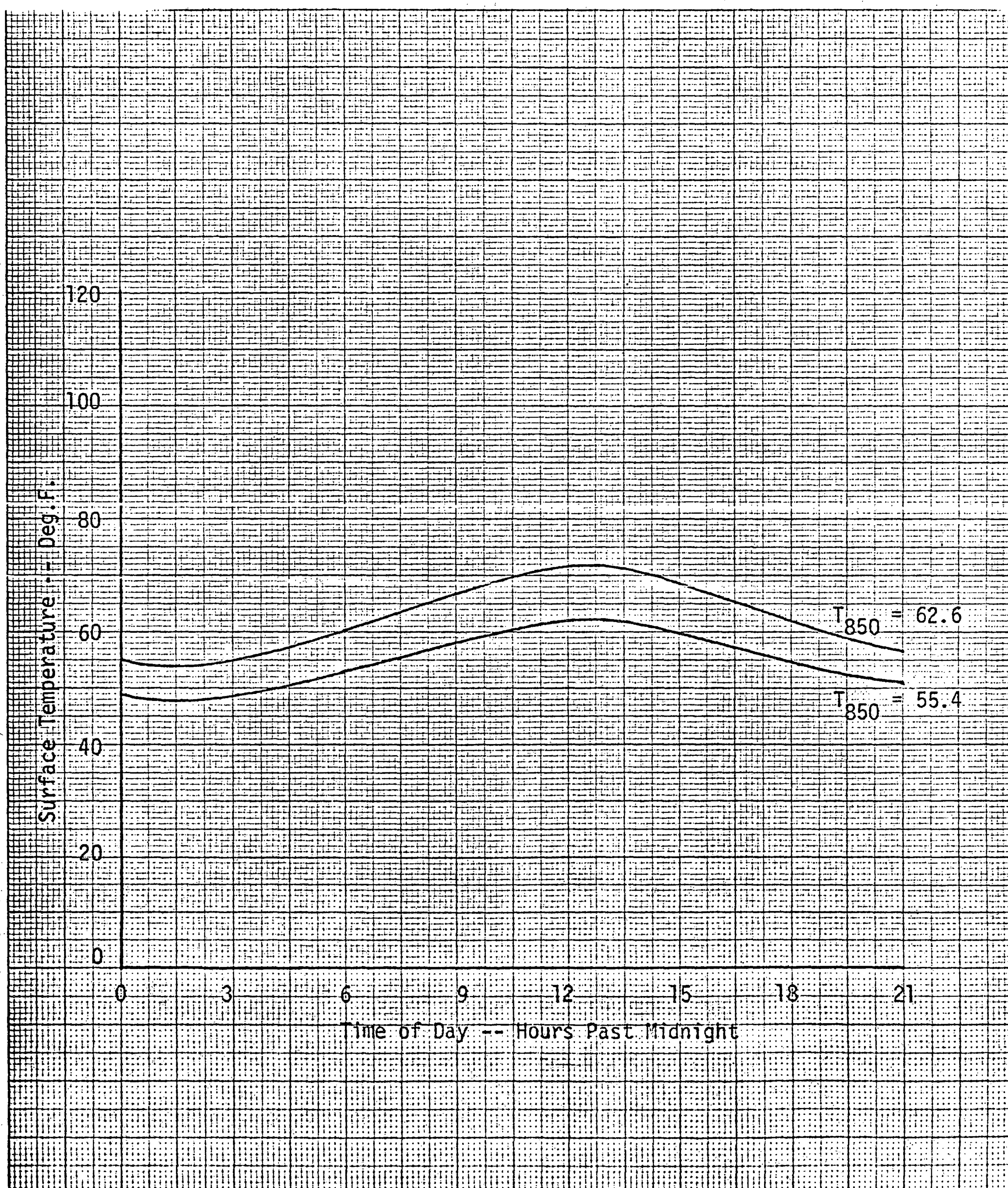


Figure 58 Surface Temperature Versus Time of Day with 850 MB Temperature as a Parameter



```
      FUNCTION FTEMP(X,Y,TOD,T850,JDOY)
C
C     A FUNCTION TO COMPUTE LOCAL SURFACE
C     TEMPERATURE GIVEN 850 MB TEMPERTURE
C
C     VER 2.00 04/21/76
C
C     PROGRAMMER - DR. JAMES SANDERLIN
C
C         INPUTS
C     X,Y     - COORDINATES OF LOCATION OF DESIRED TEMP. DATA
C     TOD     - TIME OF DAY (MIN. FROM MIDNIGHT)
C     T850    - 850 MB TEMPERATURE
C     JDOY    - JULIAN DAY OF YEAR
C
C         OUTPUTS
C     FTEMP   - SURFACE TEMPERATURE (DEG. C)
C
C         ROUTINES USED
C     HEIGHT, TOPO, TIMANG, SUNANG
C
      COMMON/CNSTNT/RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
C
C     CONVERT DAY TO REAL NUMBER
C
      DAY=FLOAT(JDOY)
C
C     CONVERT TIME OF DAY TO HOURS
C
      TIM=TOD/60.0
C
C     GET ASPECT AND ELEVATION
C
      CALL TOPO(3,X,Y,GRNG,ELEV,SRUF,SLPE,SLPN,ASPECT,SLPA)
C
C     CONVERT ASPECT TO PRINCIPAL VALUE
      A=ASPECT
      IF(A .LE. 0.0) A=A+TWOPI
      IF(A .GT. TWOPI) A=A-TWOPI
C
C     GET LOCAL HOUR ANGLE
C
      Q=(0.7*SIN(-0.986*DAY*PIO180)+SIN(PIO180*(-1.9715*DAY
     &          -15.78)))/6.0
C
C     SET LOCAL TIME DIFFERENCE FROM STANDARD MERIDIAN TO
C      ZERO
C
      DTL=0.0
```

40.0 HOUR ANGLE (OURANG) Version 1.00 8 April 1976

1. PURPOSE

This function calculates local hour angle for specified day of year and time of day.

2. ARGUMENTS

INPUT

DAY	=	number of days since 31 December
TIM	=	number of hours since midnight

OUTPUT

OURANG	=	local hour angle with respect to meridian (rad)

3. PROCEDURE

The time equation is given by

$$Q = (0.7 \sin(-0.9863D) + \sin(-1.9715D - 15.78))/6$$

where D = day of year. The local hour angle with respect to the meridian is given by

$$h = 15\ (t + Q) - 180$$

where t = time of day.

4. MODEL RESULTS

Figure 59 shows hour angle plotted against time of day for two days of the year as calculated by OURANG.

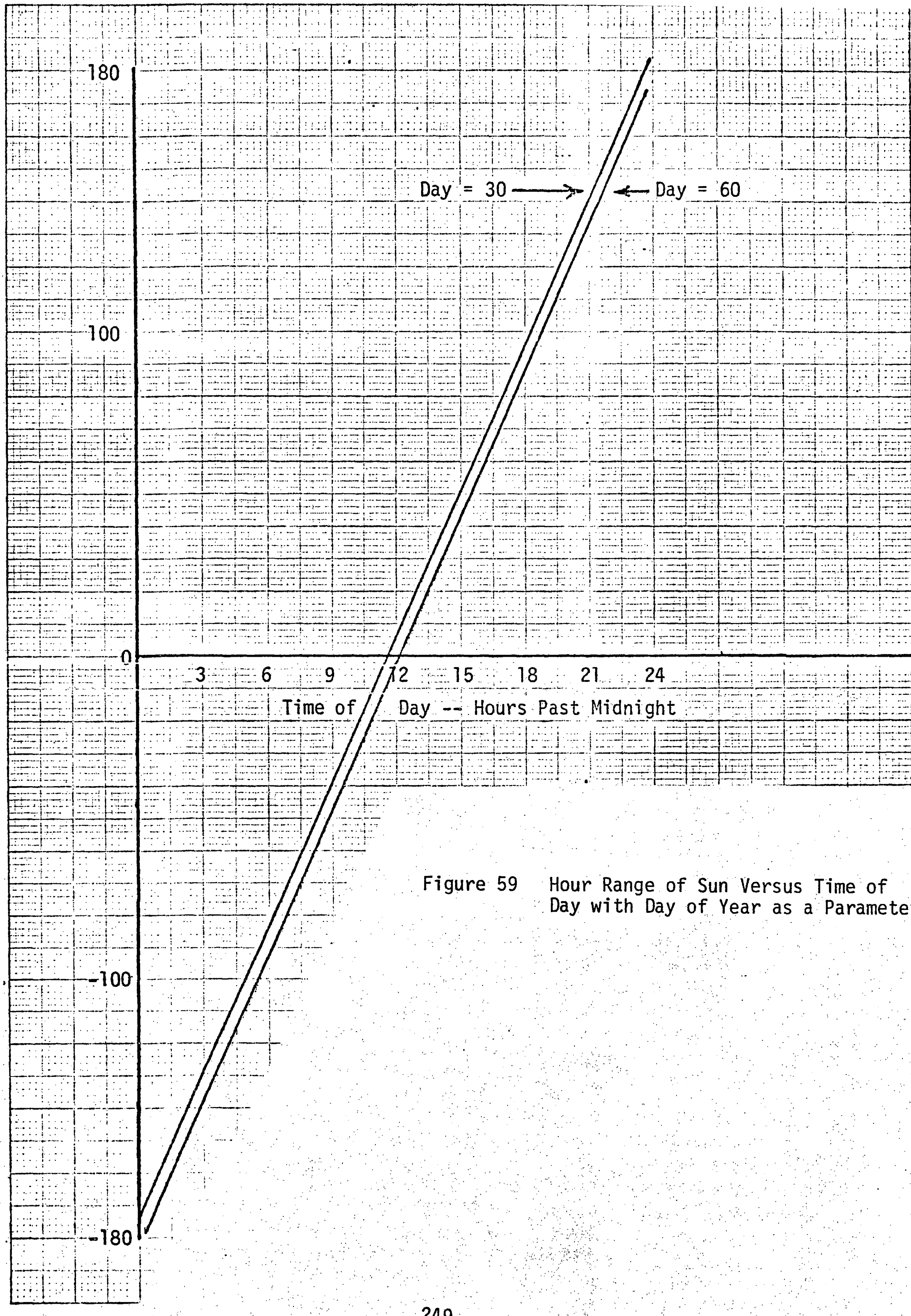


Figure 59 Hour Range of Sun Versus Time of Day with Day of Year as a Parameter

```
C
C     HOUR ANGLE WITH ONE HOUR RETARDATION
C
      HA=PIO180*(15.0*(TIM+Q+DTL)-180.0)
C
C     GET ANGLES OF SUN FROM VERTICAL AND NORTH
C
      CALL SUNANG(PIO180,DAY,HA,AV,AN)
C
C     COSINE OF ANGLE BETWEEN SURFACE AND RADIATION
C
      CALL INSOLA(HALFPI,SLPA,A,AV,AN,COSPHI)
C
C     INSOLATION COEFFICIENT (DAYTIME ONLY)
C
      ENSOL=1.0+COSPHI
      IF(AV .GE. HALFPI)ENSOL=1.0
C
C     DIURNAL VARIATION
C
      DIVAR=0.91+0.13*SIN(0.2618*TIM-2.0944)
C
C     LAPSE TEMPERATURE AT HEIGHT OF LOCATION
C
      HEIGH=HEIGHT(X,Y)
      TEMPH=T850-6.48E-3*(HEIGH-1650.0)
C
C     SURFACE TEMPERATURE
C
      FTEMP=ENSOL*DIVAR*TEMPH
      RETURN
      END
```

41.0 PRINT PERIMETER TABLE (PTABL1)

1. Purpose

This routine prints out a table of data associated with the evaluate initial attack perimeters.

2. Arguments

Input

TOF	-	time of fire report (min. from midnight)
JDOF	-	day of fire (days from 31 December)
XF,YF	-	fire ignition point coordinates
PT	-	perimeter time array (in min. from TOF)
PX,PY	-	perimeter points array
NPT	-	number of perimeters
NXY	-	number of perimeter points per perimeter

Output

A table of data is printed.

3. Procedure

The area and perimeter lengths for each initial perimeter are computed. These computations assume elliptic perimeters. That is, the semi-major axis (a) semi-minor axis (b) of each elliptic perimeter is obtained by calling the routine ABANGL. The area and perimeter are obtained from

$$\text{Area} = \pi ab,$$

$$\text{Perimeter} \approx 2\pi \sqrt{(a^2 + b^2)/2.0}.$$

The computed values are converted to acres and miles respectively. The angle of orientation is also printed out in degrees together with the 24-hour clock time and date of each perimeter.

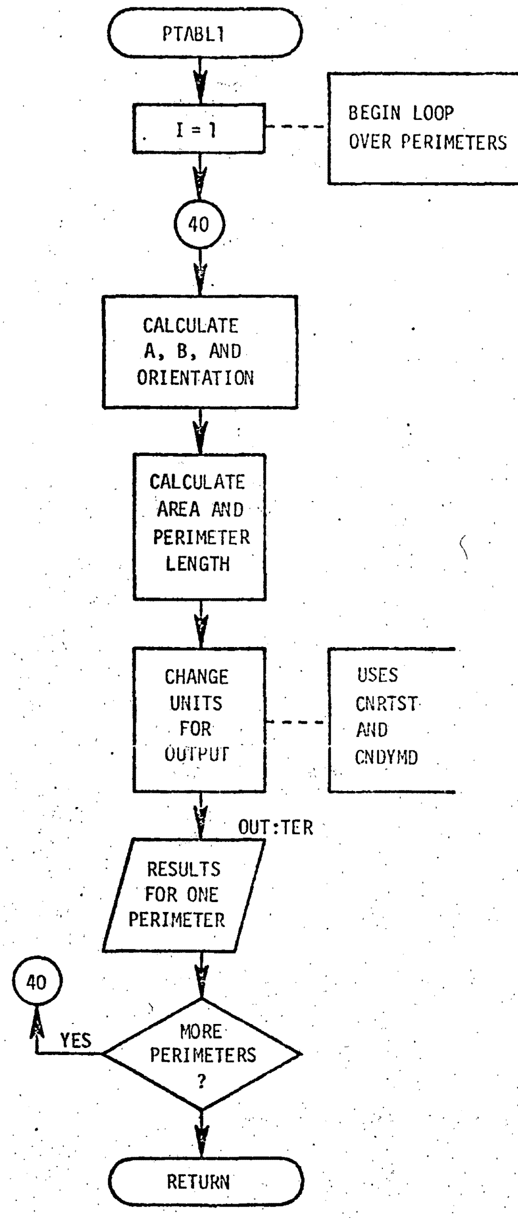
4. Comments

This routine is included for use in testing. A command for calculating and tabulating these data will replace this routine in the operational system.

5. Flow Chart

See Figure 60.

Figure 60 Print Perimeter Table



```
$$$$$PTABL1
10000         $ LINK UFILE4
10001         SUBROUTINE FILE4
10002C        PRINT,"FILE 4 NOW LOADED"
10003         RETURN
10004         END
10010         SUBROUTINE PTABL1(IYEAR)
10020C
10030C        THIS SUBROUTINE PRINTS TABULAR DATA FOR INITIAL PERIMETERS
10040C        VER 2.00
10050C
10060C        PROGRAMMER - JOHN SUNDERSON, JR.
10070C
10080C            INPUTS FROM COMMON /PERIM1/
10090C        NPT      - NUMBER OF INITIAL ATTACK PERIMETERS
10100C            INPUTS FROM COMMON /CNSTNT/
10110C        PI       - 3.14159...
10120C        TWOPI    - 6.28...
10130C        PIO180   - 0.01745... (PI/180)
10140C            INPUTS FROM COMMON /FIRE1/
10150C        XF       - X COORDINATE OF FIRE IGNITION
10160C        YF       - Y COORDINATE OF FIRE IGNITION
10170C            OUTPUT
10180C        A TABLE IS PRINTED ON THE USER TERMINAL WHICH GIVES A TIME
10190C                 HISTORY OF CALCULATED SIZE
10200C            ROUTINES USED
10210C        ABANGL,CNRTST,CNDYMD,SQRT-SYS,IFIX-SYS,FLOAT-SYS
10220C
10230         COMMON/TIME1/JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
10240         COMMON/CNSTNT/RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
10250         COMMON/FIRE1/XF,YF,AREAI,PTIN(11),NPTRQ,PTCAL(11),NPTCAL,IDISP
10260         COMMON/PERIM1/NPT,NXY,DELT,PT(11),PX(20,11),PY(20,11)
10270C
10280C       PRINT PERIMETER RESULTS
10290C
10291         XFF=XF*3.280839895
10292         YFF=YF*3.280839895
10300         PRINT   1000,XFF,YFF
10310 1000    FORMAT(/20X,30HINITIAL ATTACK PERIMETER TABLE/3X,
10320        & 31HFIRE IGNITION COORDINATES, X = ,F7.0,6H, Y = ,F7.0/2X,
10330        & 61H PER.   DATE     TIME      AREA    LENGTH     A        B    DIR./
10340        & 2X,
```

```
     & 62H  NO.                       (ACRES)  (MILES) (MILES) (MILES) (DEG)
     & /)
C
C      COMPUTE AND PRINT DATA FOR EACH PERIMETER
C
       I=1
       CMILE=0.0006213711
C          COMPUTE A AND B FOR FIRST PERIMETER
       CALL ABANGL(1,A,B,ANG)
C          COMPUTE AREA FOR FIRST PERIMETER
       AREA=PI*A*B*2.471E-4
C          CHANGE UNITS
       A=A*CMILE
       B=B*CMILE
C          COMPUTE PERIMETER LENGTH
       PRMLN=TWOPI*SQRT((A*A+B*B)*0.5)
C          CHANGE UNITS
       CALL CNRTST(TOF,JDOF,IYEAR,PT(1),TIME,IDAY,IYEARP)
       ITHRS=IFIX((TIME+0.1)/60.0)
       ITMIN=IFIX(TIME+0.1)-ITHRS*60
       CALL CNDYMD(IDAY,IYEARP,IMON,IDY)
C          OUTPUT RESULTS
       IQDD1=IDY/10
       IQDD2=IDY-IQDD1*10
       IQYD1=IYEARP/10
       IQYD2=IYEARP-IQYD1*10
       IQHD1=ITHRS/10
       IQHD2=ITHRS-IQHD1*10
       IQMD1=ITMIN/10
       IQMD2=ITMIN-IQMD1*10
       PRINT     2000,I,IMON,IQDD1,IQDD2,IQYD1,IQYD2,IQHD1,IQHD2,
     & IQMD1,IQMD2,AREA,PRMLN,A,B
  2000 FORMAT(4X,I2,2X,I2,2(1H/,2I1),1X,2I1,1H:,2I1,1X,F11.2,3(1X,F7.3),
     & 4H N/A)
  3000 FORMAT(4X,I2,2X,I2,2(1H/,2I1),1X,2I1,1H:,2I1,1X,F11.2,3(1X,F7.3),
     & 1X,F5.1)
C
C      DO THE SAME FOR REMAINING PERIMETERS ADDING ORIENTATION DIRECTION
C
       DO 40 I=2,NPT
       CALL ABANGL(I,A,B,ANG)
       AREA=PI*A*B*2.471E-4
       A=A*CMILE
       B=B*CMILE
       PRMLN=TWOPI*SQRT((A*A+B*B)*0.5)
       CALL CNRTST(TOF,JDOF,IYEAR,PT(I),TIME,IDAY,IYEARP)
       ITHRS=IFIX((TIME+0.1)/60.0)
       ITMIN=IFIX(TIME+0.1)-ITHRS*60
       CALL CNDYMD(IDAY,IYEARP,IMON,IDY)
       IF(ANG.LT.0.0)ANG=ANG+TWOPI
       ADEG=ANG/PIO180
       IQDD1=IDY/10
       IQDD2=IDY-IQDD1*10
       IQYD1=IYEARP/10
       IQYD2=IYEARP-IQYD1*10
       IQHD1=ITHRS/10
       IQHD2=ITHRS-IQHD1*10
       IQMD1=ITMIN/10
       IQMD2=ITMIN-IQMD1*10
       PRINT     3000,I,IMON,IQDD1,IQDD2,IQYD1,IQYD2,IQHD1,IQHD2,
     & IQMD1,IQMD2,AREA,PRMLN,A,B,ADEG
    40 CONTINUE
C
       RETURN
       END
```

42.0 RADII AND ANGLE RETRIEVAL ROUTINE (ABANGL)

1. Purpose

This routine recovers radii and angle values from a set of x,y points computed for perimeters calculated from elliptic spread rates.

2. Arguments

<u>Input</u> - from list

I	-	the index specifying which perimeter is being considered.

<u>Input</u> - from COMMON

PX,PY	-	arrays containing the x,y coordinates of the perimeter points for initial attack
NXY	-	the number of points per perimeter
NPT	-	total number of perimeters

<u>Output</u> - to list

A	-	semi-major axis length of I^{th} perimeter (meters)
B	-	semi-minor axis length of I^{th} perimeter (meters)
ANG	-	the azimuthal angle of the semi-major axis (in radians)

3. Procedure

This routine assumes that the original x,y points were stored in a fixed format, the number of x,y points per ellipse is a multiple of four, and that each quadrant of the ellipse contains the same number of points. The points are taken to be stored in a counterclockwise order starting with the point at the head of the fire. Thus, the point at the tail of the fire is given by the index NXY/2, and these two points define the major axis length (a) while the indices defining the ends of the minor axis length (b) are NXY/4 and 3*NXY/4.

The routine tests to insure that NXY is divisible by 4. It also checks to see if the input index (I) is within the bounds defined by NPT, the number of perimeters. The indices defining the end points of the major and minor axes of the ellipse are then computed from:

$$n_1 = 1,$$

$$n_2 = n_1 + \Delta n,$$

$$n_3 = n_2 + \Delta n,$$

$$n_4 = n_3 + \Delta n$$

where

$$\Delta n = NXY/4.$$

The numerical values of a and b are computed by:

$$a_X = PX(1,I) - PX(n_3,I),$$

$$a_Y = PY(1,I) - PY(n_3,I),$$

$$b_X = PX(n_2,I) - PX(n_a.I),$$

$$b_Y = PY(n_2,I) - PY(n_4,I),$$

$$a = \frac{1}{2}\sqrt{(a_X^2 + a_Y^2)} \qquad = \quad A,$$

$$b = \frac{1}{2}\sqrt{(b_X^2 + b_Y^2)} \qquad = \quad B.$$

The angle of orientation, α, is then given by

$$\alpha = \arctan(a_Y/a_X).$$

The azimuthal angle of orientation is

$$ANG = \pi/2 - \alpha.$$

After a, b and α are computed, control is returned to the calling program.

4. Comments

None.

5. Flow Chart

See Figure 61.

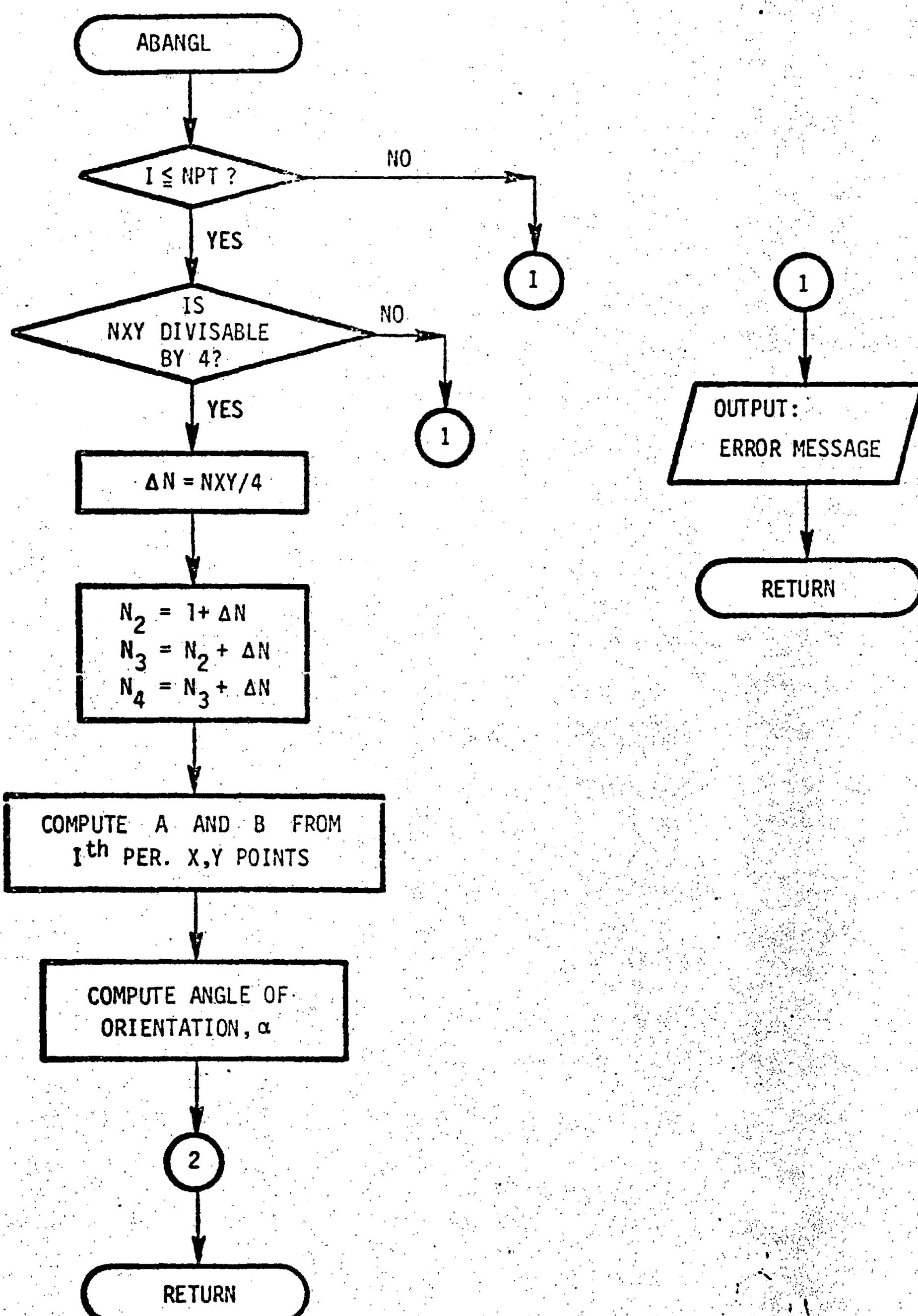


Figure 61 Logic Structure for Routine ABANGL

```
      SUBROUTINE ABANGL(I1,A,B,ANG)
C
C     THIS SUBROUTINE RECOVERS A,B, AND ANGLE VALUES FROM A SET OF
C     X,Y POINTS COMPUTED FROM ELLIPTIC SPREAD RATES
C     VERSION 1.00
C
C     PROGRAMMER - J. KEEFER
C
C          INPUTS - FROM LIST
C     I1        - THE INDEX SPECIFYING WHICH PERIMETER IS
C                 BEING CONSIDERED
C          INPUTS - FROM COMMON
C     PX,PY     - THE PERIMETER POINT ARRAYS
C     NXY       - THE NUMBER OF POINTS PER PERIMETER
C     NPT       - THE NUMBER OF PERIMETERS
C          OUTPUTS
C     A         - SEMI-MAJOR AXIS LENGTH(METERS)
C     B         - SEMI-MINOR AXIS LENGTH(METERS)
C     ANG       - THE AZIMUTHAL DIRECTION OF THE SEMI-MAJOR AXIS
C                 IN A COUNTERCLOCKWISE DIRECTION
C
C          ROUTINES USED
C     ERRREQ,ATAN2
C
      COMMON /PERIM1/ NPT,NXY,DELT,PT(11),PX(20,11),PY(20,11)
      COMMON /CNSTNT/ RE,PI,PIO180,HALFPI,TWOPI,FOURPI,ATEPI,GRAVZ
C
C     TEST FOR CORRECT DIMENSIONS
C
      I=I1
    5 IF(I1.LE.NPT) GO TO 10
      I=NPT
C
C     TEST FOR CORRECT VALUE FOR THE NUMBER OF POINTS
C
   10 CONTINUE
      JSTEP = NXY/4
C
C     COMPUTE THE POINT INDICES NEEDED FOR CALCULATING A AND B
C
   20 CONTINUE
      N2 = 1 + JSTEP
      N3 = N2 + JSTEP
      N4 = N3 + JSTEP
C
C     COMPUTE A AND B
C
      AX = PX(1,I) - PX(N3,I)
      AY = PY(1,I) - PY(N3,I)
      BX = PX(N2,I) - PX(N4,I)
      BY = PY(N2,I) - PY(N4,I)
      A = SQRT(AX*AX + AY*AY)/2.0
      B = SQRT(BX*BX + BY*BY)/2.0
C
C     COMPUTE ANGLE
C
      ANG = ATAN2(AY,AX)
      IF(ANG.LT.0.0)  ANG = ANG + TWOPI
      ANG = HALFPI - ANG
C
      RETURN
      END
```

43.0 EVALUATE INITIAL CONTROL (EVLCØN)

1. Purpose

This routine calculates growth rates and probability approximations.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

For each perimeter time, the work accomplished by each resource which has arrived is calculated. Work rates and growth rates are then calculated using a discrete derivative (see DERIV). These values are then used to calculate the probabilities of control and containment by the following equations:

$$P_{CNT_i} = W_{D_i}/P_{L_i}$$

$$P_{CNR_i} = P_{CNT_i} \cdot C_{WR_i}/P_{GR_i}$$

where

P_{CNT_i} = approximate probability of containment

P_{CNR_i} = approximate probability of control

W_{D_i} = work done

P_{L_i} = perimeter length

C_{WR_i} = cumulative work rate

P_{GR_i} = perimeter growth rate

i = ith time of calculation

Results are then output to the user under interactive control by the user.

4. Comments

The listings which follow the flow chart are of two different programs (EVLP1, EVLP2). The EVLCØN described here was broken into these two parts for overlay purposes. Logically the two parts should be viewed as a unit structure.

5. Flow Chart

See Figure 62.

Figure 62 Evaluate Control

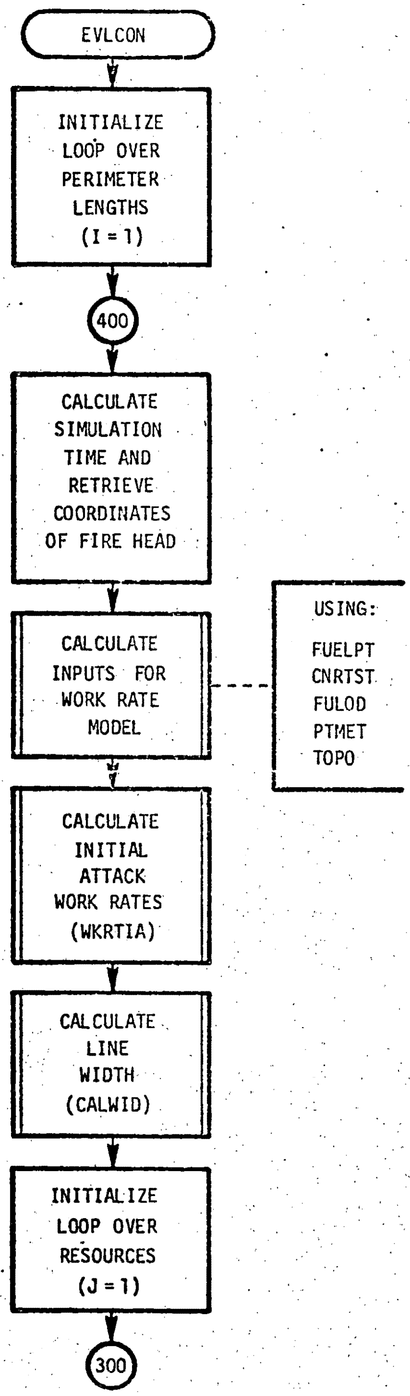


Figure 62 Evaluate Control (Cont'd)

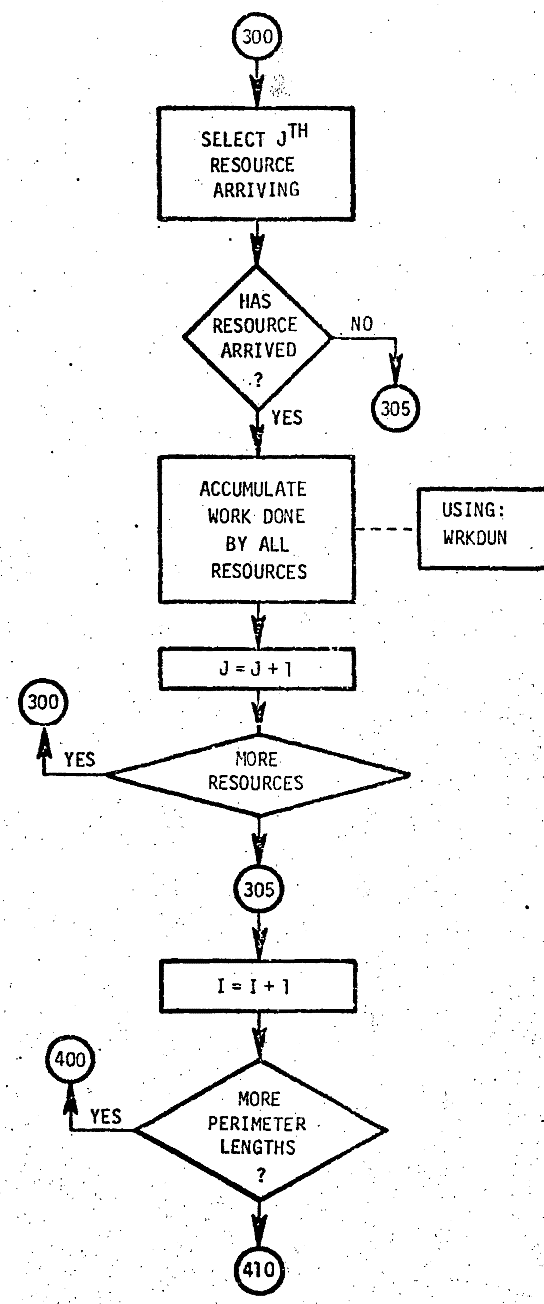


Figure 62 Evaluate Control (Cont'd)

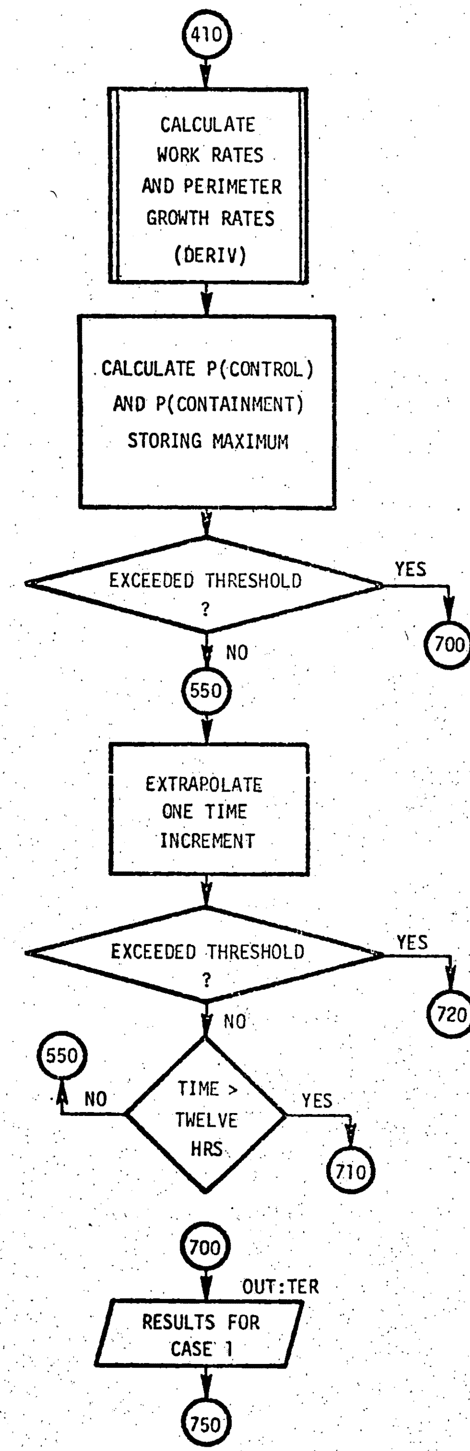
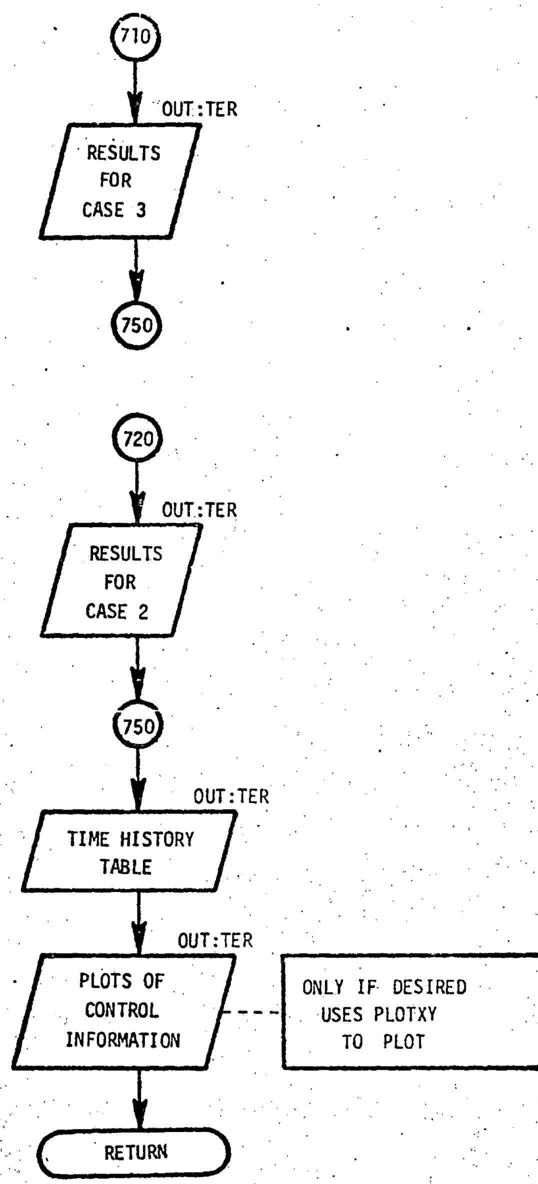


Figure 62 Evaluate Control (Cont'd)



```
$$$$$SEVLP1
      $ LINK UEVLP1
      SUBROUTINE FEVLP1
C     PRINT,"FILE UEVLP1 LOADED"
      RETURN
      END
      SUBROUTINE EVLP1(IYRF,WATERA,DISORI)
C
C     THIS ROUTINE EVALUATES AND OUTPUTS THE SUCCESS
C     OF AN INITIAL ATTACK
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS FROM LIST
C     IYRF    - YEAR OF FIRE (YEARS-1900)
C     WATERA  - WATER AVAILABILITY (ALPHA,YES=WATER AT SITE)
C         INPUTS FROM COM ON /TIME1/
C     JDOF    - DAY OF FIRE
C     TOF     - TIME OF FIRE
C         INPUTS FROM COMMON /FIRE1/
C     XF      - X COORDINATE OF FIRE IGNITION
C     YF      - Y COORDINATE OF FIRE IGNITION
C     NPTRQ   - NUMBER OF PERIMETER TIMES REQUESTED
C         INPUTS FROM COMMON /IMPPER/
C     TINCR   - FINE SCALE TIME INCREMENT
C     NTSTEP  - NUMBER OF FINE SCALE TIME STEPS
C     PL      - AN ARRAY OF PERIMETER LENGTHS (SIZE=NTSTEP)
C         INPUTS FROM COMMON /ROUT12/
C     IORD    - AN ARRAY OF INDIRECT SUBSCRIPTS FOR RESOURCES IN
C               ORDER OF INCREASING ETA (SIZE=NRSTOT)
C     ETA     - AN ARRAY OF ETAS (SIZE=NRSTOT)
C     IDR     - AN ARRAY OF INDICES ON THE RESOURCE CHARACTERISTICS
C               LIST (SIZE=NRSTOT)
C     ACT     - AN ARRAY OF AIR CYCLE TIMES (SIZE=NRSTOT)
C         INPUTS FROM COMMON /SPLMET/
C     JDLRF   - DAY OF LAST RAINFALL
C
C         OUTPUTS TO COMMON /IMPPER/
C     WD      - AN ARRAY OF CUMULATIVE WORK DONE (SIZE=NTSTEP)
C     PCNR    - AN ARRAY OF PROBABILITY OF CONTROL (SIZE=NTSTEP)
C     PCNT    - AN ARRAY OF PROBABILITY OF CONTAINMENT(SIZE=NTSTEP)
C     PGR     - AN ARRAY OF PERIMETER GROWTH RATE (SIZE=NTSTEP-1)
C     CWR     - AN ARRAY OF CUMULATIVE WORK RATES (SIZE=NTSTEP-1)
C
      ALPHA CHAR(102)
      ALPHA WATERA,YES,NO,ANS,ZER,PLUS
C
      DIMENSION XCHD(51),YCHD(51)
      DIMENSION WRRES(15)
C
      COMMON/TIME1/JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
      COMMON/FIRE1/XF,YF,AREA1,PTIN(11),NPTRQ,PTCAL(11),NPTCAL,IDISP
      COMMON /IMPPER/ TINCR,NTSTEP,PL(51),WD(51),PCNR(51),PCNT(51),
     &                PGR(51),CWR(51)
      COMMON/ROUT12/NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
     &              ETA(20),ACT(20)
      COMMON/SPLMET/IYLR,JDLRF,DRAIN,IYRM,IDOM,TDOM
C
      EQUIVALENCE (XCHD(1),PCNR(1)),(YCHD(1),PCNT(1))
C
      DATA YES /3HYES/,NO/2HNO/,ZER/1H0/,PLUS/1H+/
      DATA PCNRTH/0.95/
      DATA TWLHRS/420.0/
      DATA NRES/0/
C
      PRINT    2005
 2005 FORMAT(/20X,20HSUPPRESSION ANALYSIS)
C
C     LOOP OVER TIMES AT WHICH PERIMETER LENGTHS HAVE BEEN
C     CALCULATED
```

```
C
      DO 400 I=1,NTSTEP
C
C     SIMTIM IS A TIME IN MINUTES RELATIVE TO THE TIME OF THE FIRE
C
      SIMTIM = TINCR*FLOAT(I-1)
C         CALCULATE COORDINATES OF POINT AT HEAD OF FIRE
C               THIS POINT WILL BE USED TO DETERMINE PHYSICAL
C               CHARACTERISTICS (MET,FUEL,SLOPE,ETC.)
      XH=XCHD(I)
      YH=YCHD(I)
C
C     GET PHYSICAL CHARACTERISTICS
C
C         DISTANCE OFF ROAD = DISTANCE OFF ROAD INPUT + SQRT(DISTANCE TO
C               HEAD OF FIRE)
      DISTOR=DISORI+SQRT(SQRT((XH-XF)**2+(YH-YF)**2))
C         FUEL TYPE AT HEAD
      CALL FUELPT(XH,YH,IFTYP,DUM,AGE,DUM2)
C         FUEL LOAD AT HEAD
      CALL CNRTST(TOF,JDOF,IYRF,SIMTIM,TIMN,NDAY,NYR)
C         DAY - DAYS SINCE MAY 1 (SINCE APRIL 30 ON LEAP YEAR)
      DAY=FLOAT(NDAY-121)
      IF(DAY.LE.0.0.AND.JDLRF.LE.NDAY)DAY=FLOAT(NDAY-JDLRF)
      IF(JDLRF.GT.NDAY)DAY=FLOAT(365-JDLRF+NDAY)
      CALL FULOD(IFTYP,AGE,DAY,FLOD)
C         WIND SPEED (METERS/MIN) AT HEAD
      CALL PTMET(XH,YH,TIMN,NDAY,WAS,WVEL,SDWAS,SDWVEL,HREL,TEMP)
C         SLOPE AND SURFACE ROUGHNESS
      CALL TOPO(4,XH,YH,BRNG,ZH,SRUF,SLE,SLN,ASP,SLOPE)
C
C     CALCULATE WORK RATES MODIFIED FOR ENVIRONMENT
C
      CALL WKRTIA(FLOD,WVEL,SLOPE,SRUF,TEMP,DISTOR,WRRES)
C
C     CALCULATE WIDTH OF LINE
C
      CALL CALWID(FLOD,WVEL,WIDTH)
C         INITIALIZE WORK DONE AT THIS TIME (WD)
      WD(I)=0.0
C
C     LOOP OVER RESOURCES
C
      DO 300 J=1,NRSTOT
C         SELECT RESOURCES IN ORDER OF INCREASING ETA
      KRES=IORD(J)
C
C     HAS THIS RESOURCE AND ALL FOLLOWING RESOURCES ARRIVED
C
      IF(ETA(KRES).GE.SIMTIM)GO TO 305
C         RESOURCE HAS ARRIVED AT FIRE
      TF=SIMTIM-ETA(KRES)
      TI=AMAX1(0.0,TF-TINCR)
C         ASSIGN A LARGE VOLUME OF WATER IF WATER IS AT SITE
      WATER=0.0
      IF(WATERA.EQ.YES)WATER=1.0E20
C
C     ACCUMULATE WORK DONE BY ALL RESOURCES WHICH HAVE
C     ARRIVED BY THIS SIMTIM
C
      WD(I)=WD(I)+WRKOUN(IDR(KRES),TI,TF,ACT(KRES),
     &        WATER,WIDTH,WRRES,NRES)*FLOAT(NORS(KRES))
C         END OF LOOP OVER AVAILABLE RESOURCES
  300 CONTINUE
C         NO MORE RESOURCES AVAILABLE
  305 CONTINUE
```

```
      IF(I.GT.1)WD(I)=WD(I)+WD(I-1)
C          END OF LOOP OVER FINE TIME STEPS
  400 CONTINUE
C
C     CALCULATE DERIVATIVES OF WORK DONE AND PERIMETER LENGTH
C
      CALL DERIV(TINCR,0.0,WD,NTSTEP,CWR,X1CWR)
      CALL DERIV(TINCR,0.0,PL,NTSTEP,PGR,X1PGR)
C
C     CALCULATE PROBABILITY OF CONTROL AND CONTAINMENT
C     (STORING MAX. P(CONTROL))
C
      NTSTEP=NTSTEP-1
      CRMAX=0.0
      CNMAX=0.0
      CTMAX=0.0
      IPCNRG=0
      DO 500 I=1,NTSTEP
      PCNT(I)=0.99
      IF(PL(I).LE.0.0)GO TO 410
      PCNT(I)=AMAX1(0.01,AMIN1(WD(I)/PL(I), 0.99 ))
  410 PCNR(I)=0.99*PCNT(I)
      IF(PGR(I).LE.0.0)GO TO 420
      PCNR(I)=AMAX1(0.01 ,AMIN1(CWR(I)/PGR(I), 0.99 ))*PCNT(I)
      PCNR(I)=AMAX1(0.01,AMIN1(0.99,PCNR(I)))
  420 IF(PCNR(I).GT.PCNRTH.AND.IPCNRG.EQ.0)IPCNRG=I
      CALL MAXCR(PCNR(I),PCNT(I),TINCR*FLOAT(I-1),CRMAX,CNMAX,CTMAX)
  500 CONTINUE
      NTSTEP=NTSTEP+1
      IF(IPCNRG.GT.0)GO TO 700
      IPCNRG=NTSTEP-1
C
C     NO CALCULATED PROBABILITY OF CONTROL GREATER THAN THRESHOLD
C     EXTRAPOLATE UNTIL 12 HRS PAST TOF
C
      PCNRN=PCNR(NTSTEP-1)
      SIMTIM=TINCR*FLOAT(NTSTEP-1)
      IF(SIMTIM.GE.TWLHRS)GO TO 710
C          WDONL IS WORK DONE LAST
      WDONL=WD(NTSTEP)
C          IEXTR IS THE EXTRAPOLATION STEP INDEX
      IEXTR=1
C          TEXTR IS AMOUNT OF TIME EXTRAPOLATED
      TEXTR=TINCR
C          SIMTIM IS SIMULATION TIME (RELATIVE TO TOF)
      SIMTIM=SIMTIM+TEXTR
      SIMTB=SIMTIM
C          PGRT IS PERIMETER GROWTH RATE (ASSUMMED CONSTANT FOR EXTRAP.)
      PGRT=PGR(NTSTEP-1)
C          PLEN IS PERIMETER LENGTH
      PLEN=PL(NTSTEP)
      PLEN=PLEN+PGRT*TINCR
      PCNTL=PCNT(NTSTEP-1)
      PCNRL=PCNR(NTSTEP-1)
C          WDON IS WORK DONE
  550 WDON=0.0
C
C     LOOP OVER RESOURCES
C
      DO 600 J=1,NRSTOT
C          SELECT RESOURCES IN ORDER OF INCREASING ETA
      KRES=IORD(J)
C
C     HAS THIS RESOURCE AND ALL FOLLOWING RESOURCES ARRIVED
C
      IF(ETA(KRES).GE.SIMTIM)GO TO 605
```

```
C        RESOURCE HAS ARRIVED AT FIRE
      TF=SIMTIM-ETA(KRES)
      TI=AMAX1(0.0,TF-TINCR)
C        ASSIGN A LARGE VOLUME OF WATER IF WATER IS AT SITE
      WATER=0.0
      IF(WATERA.EQ.YES)WATER=1.0E20
C
C     ACCUMULATE WORK DONE BY ALL RESOURCES WHICH HAVE
C     ARRIVED BY THIS SIMTIM
C
      WDON=WDON+WRKDUN(IDR(KRES),TI,TF,ACT(KRES),
     &       WATER,WIDTH,WRRES,NRES)*FLOAT(NORS(KRES))
C        END OF LOOP OVER AVAILABLE RESOURCES
  600 CONTINUE
C        NO MORE RESOURCES AVAILABLE
  605 CONTINUE
      WDON=WDON+WDONL
      PCNTN=AMAX1(0.01 ,AMIN1(WDON/PLEN, 0.99))
      PCNRN=AMAX1(0.01 ,AMIN1((WDON-WDONL)/TINCR/PGRT, 0.99 ))*PCNTN
      PCNRN=AMAX1(0.01,AMIN1(0.99,PCNRN))
      IF(PCNRN.GT.PCNRTH)GO TO 720
      CALL MAXCR(PCNRN,PCNTN,SIMTIM,CRMAX,CNMAX,CTMAX)
C        START NEXT EXTRAPOLATION
      IEXTR=IEXTR+1
      SIMTIM=SIMTB+TINCR*FLOAT(IEXTR-1)
      IF(SIMTIM.GE.TWLHRS)GO TO 710
      PCNTL=PCNTN
      PCNRL=PCNRN
      WDONL=WDON
      PLEN=PLEN+PGRT*TINCR
      GO TO 550
C
C     PROBABILITY OF CONTROL INDEX FOUND, PRINT RESULTS
C
  700 CALL OUTTIM(TOF,JDOF,IYRF,TINCR*FLOAT(IPCNRG-1),
     & MONC,JDAYC,IYRC,IHRSC,IMINC)
C
      ITCN=IFIX(PCNT(IPCNRG)*100.0+0.5)
      ITCR=IFIX(PCNR(IPCNRG)*100.0+0.5)
C
      IQDD1=JDAYC/10
      IQDD2=JDAYC-IQDD1*10
      IQYD1=IYRC/10
      IQYD2=IYRC-IQYD1*10
      IQHD1=IHRSC/10
      IQHD2=IHRSC-IQHD1*10
      IQMD1=IMINC/10
      IQMD2=IMINC-IQMD1*10
      IQND1=ITCN/10
      IQND2=ITCN-IQND1*10
      IQRD1=ITCR/10
      IQRD2=ITCR-IQRD1*10
      PRINT    2000,MONC,IQDD1,IQDD2,IQYD1,IQYD2,
     & IQHD1,IQHD2,IQMD1,IQMD2,IQND1,IQND2,
     & IQRD1,IQRD2
 2000 FORMAT(3X,29HCONTROL EXPECTED TO OCCUR ON ,I2,1H/,2I1,1H/,2I1,
     & 4H AT ,2I1,1H:,2I1 /
     & 3X,50HPROBABILITY OF CONTAINMENT IS ESTIMATED TO BE = 0.,2I1/
     & 3X,50HPROBABILITY OF CONTROL     IS ESTIMATED TO BE = 0.,2I1)
      GO TO 750
C
C     *** NO CONTROL PREDICTED ***
C
  710 PRINT    2001
 2001 FORMAT(3X,36H*** NO CONTROL EXPECTED TO OCCUR ***)
C
```

```
      CALL OUTTIM(TOF,JDOF,IYRF,CTMAX,MONC,JDAYC,IYRC,IHRSC,IMINC)
      ITCN=IFIX(CNMAX*100.0+0.5)
      ITCR=IFIX(CRMAX*100.0+0.5)
      IQDD1=JDAYC/10
      IQDD2=JDAYC-IQDD1*10
      IQYD1=IYRC/10
      IQYD2=IYRC-IQYD1*10
      IQHD1=IHRSC/10
      IQHD2=IHRSC-IQHD1*10
      IQMD1=IMINC/10
      IQMD2=IMINC-IQMD1*10
      IQND1=ITCN/10
      IQND2=ITCN-IQND1*10
      IQRD1=ITCR/10
      IQRD2=ITCR-IQRD1*10
      PRINT     2002,IQRD1,IQRD2,MONC,IQDD1,IQDD2,
     & IQYD1,IQYD2,IQHD1,IQHD2,IQMD1,IQMD2,IQND1,IQND2
 2002 FORMAT(3X,54HMAXIMUM PROBABILITY OF CONTROL IS ESTIMATED TO BE = 0
     &.,2I1 /
     & 8X,13HOCCURRING ON ,I2,1H/,2I1,1H/,2I1,4H AT ,2I1,1H:,2I1 /
     & 8X,55H(ESTIMATED PROBABILITY OF CONTAINMENT AT THAT TIME = 0.,
     & 2I1,1H))
C
      CALL OUTTIM(TOF,JDOF,IYRF,SIMTIM,MONC,JDAYC,IYRC,IHRSC,IMINC)
      ITCN=IFIX(PCNTN*100.0+0.5)
      ITCR=IFIX(PCNRN*100.0+0.5)
      IQDD1=JDAYC/10
      IQDD2=JDAYC-IQDD1*10
      IQYD1=IYRC/10
      IQYD2=IYRC-IQYD1*10
      IQHD1=IHRSC/10
      IQHD2=IHRSC-IQHD1*10
      IQMD1=IMINC/10
      IQMD2=IMINC-IQMD1*10
      IQND1=ITCN/10
      IQND2=ITCN-IQND1*10
      IQRD1=ITCR/10
      IQRD2=ITCR-IQRD1*10
      PRINT     2003,IQRD1,IQRD2,MONC,IQDD1,IQDD2,
     & IQYD1,IQYD2,IQHD1,IQHD2,IQMD1,IQMD2,IQND1,IQND2
 2003 FORMAT(3X,58HMOST LIKELY PROBABILITY OF CONTROL IS ESTIMATED TO BE
     & = 0.,2I1 /
     & 8X,13H,OCCURRING ON ,I2,1H/,2I1,1H/,2I1,4H AT ,2I1,1H:,2I1 /
     & 8X,55H(ESTIMATED PROBABILITY OF CONTAINMENT AT THAT TIME = 0.,
     & 2I1,1H)/)
      GO TO 750
C
C     PROBABILITY OF CONTROL FOUND BY EXTRAPOLATION
C
  720 CALL OUTTIM(TOF,JDOF,IYRF,SIMTIM,MONC,JDAYC,IYRC,IHRSC,IMINC)
      ITCN=IFIX(PCNTN*100.0+0.5)
      ITCR=IFIX(PCNRN*100.0+0.5)
      IQDD1=JDAYC/10
      IQDD2=JDAYC-IQDD1*10
      IQYD1=IYRC/10
      IQYD2=IYRC-IQYD1*10
      IQHD1=IHRSC/10
      IQHD2=IHRSC-IQHD1*10
      IQMD1=IMINC/10
      IQMD2=IMINC-IQMD1*10
      IQND1=ITCN/10
      IQND2=ITCN-IQND1*10
      IQRD1=ITCR/10
      IQRD2=ITCR-IQRD1*10
      PRINT     2004,MONC,IQDD1,IQDD2,IQYD1,IQYD2,
     & IQHD1,IQHD2,IQMD1,IQMD2,IQND1,IQND2,
```

```
     & IQRD1,IQRD2
2004 FORMAT(3X,29HCONTROL EXPECTED TO OCCUR ON ,I2,1H/,2I1,1H/,2I1,
     & 4H AT ,2I1,1H:,2I1 /
     & 4X,39H(PROBABILITIES HAVE BEEN EXTRAPOLATED.)/
     & 3X,50HPROBABILITY OF CONTAINMENT IS ESTIMATED TO BE = 0.,2I1 /
     & 3X,50HPROBABILITY OF CONTROL     IS ESTIMATED TO BE = 0.,2I1 /)
 750 CONTINUE
     RETURN
     END
```

$$$$$SEVLP2

```
      $ LINK UEVLP2
      SUBROUTINE FEVLP2
C     PRINT,"FILE UEVLP2 NOW LOADED"
      RETURN
      END
       OPTION NOCHECK,NOLINE
      SUBROUTINE EVLP2(IYRF,WATERA)
C
C     THIS ROUTINE EVALUATES AND OUTPUTS THE SUCCESS
C     OF AN INITIAL ATTACK
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS FROM LIST
C     IYRF   - YEAR OF FIRE (YEARS-1900)
C     WATERA - WATER AVAILABILITY (ALPHA,YES=WATER AT SITE)
C         INPUTS FROM COM ON /TIME1/
C     JDOF   - DAY OF FIRE
C     TOF    - TIME OF FIRE
C         INPUTS FROM COMMON /FIPE1/
C     XF     - X COORDINATE OF FIRE IGNITION
C     YF     - Y COORDINATE OF FIRE IGNITION
C     NPTRQ  - NUMBER OF PERIMETER TIMES REQUESTED
C         INPUTS FROM COMMON /IMPPER/
C     TINCR  - FINE SCALE TIME INCREMENT
C     NTSTEP - NUMBER OF FINE SCALE TIME STEPS
C     PL     - AN ARRAY OF PERIMETER LENGTHS (SIZE=NTSTEP)
C         INPUTS FROM COMMON /ROUT12/
C     IORD   - AN ARRAY OF INDIRECT SUBSCRIPTS FOR RESOURCES IN
C              ORDER OF INCREASING ETA (SIZE=NRSTOT)
C     ETA    - AN ARRAY OF ETAS (SIZE=NRSTOT)
C     IDR    - AN ARRAY OF INDICES ON THE RESOURCE CHARACTERISTICS
C              LIST (SIZE=NRSTOT)
C     ACT    - AN ARRAY OF AIR CYCLE TIMES (SIZE=NRSTOT)
C         INPUTS FROM COMMON /SPLMET/
C     JDLRF  - DAY OF LAST RAINFALL
C
C         OUTPUTS TO COMMON /IMPPER/
C     WD     - AN ARRAY OF CUMULATIVE WORK DONE (SIZE=NTSTEP)
C     PCNR   - AN ARRAY OF PROBABILITY OF CONTROL (SIZE=NTSTEP)
C     PCNT   - AN ARRAY OF PROBABILITY OF CONTAINMENT(SIZE=NTSTEP)
```

```
C     PGR    - AN ARRAY OF PERIMETER GROWTH RATE (SIZE=NTSTEP-1)
C     CWR    - AN ARRAY OF CUMULATIVE WORK RATES (SIZE=NTSTEP-1)
C
      ALPHA CHAR(102)
      ALPHA AST,AT
      ALPHA WATERA,YES,NO,ANS,ZER,PLUS
C
      DIMENSION PLTX(104),PLTY(104)
      DIMENSION XCHD(51),YCHD(51)
      DIMENSION WHRES(15)
C
      COMMON/TIME1/JDOF,TOF,JDOS,TOS,JDMAX,TODMAX,REAL
      COMMON/FIRE1/XF,YF,AREA1,PTIN(11),NPTRQ,PTCAL(11),NPTCAL,IDISP
      COMMON /IMPPER/ TINCR,NTSTEP,PL(51),WD(51),PCNR(51),PCNT(51),
     &                PGR(51),CWR(51)
      COMMON/ROUT12/NRSTOT,NRSMAX,IORD(20),IDR(20),NORS(20),
     &               ETA(20),ACT(20)
      COMMON/SPLMET/IYER,JDLRF,DRAIN,IYRM,IDOM,TDOM
C
      EQUIVALENCE (XCHD(1),PCNR(1)),(YCHD(1),PCNT(1))
C
      DATA YES /3HYES/,NO/2HNO/,ZER/1H0/,PLUS/1H+/
      DATA PCNRTH/0.95/
      DATA TWLHRS/420.0/
      DATA AST/1H*/,AT/1H@/
C         CALCULATE TIME ARRAY FOR USE IN PLOTTING
      NM1=NTSTEP-1
      LAST=NM1*2
      DO 790 I=1,NM1
      SIMTIM=TINCR*FLOAT(I-1)
      PLTX(I)=SIMTIM
      PLTX(I+NM1)=SIMTIM
C         CONVERT PERIMETER AND CONTROL LINE LENGTHS TO MILES
      PL(I)=PL(I)*0.00062136362
      WD(I)=WD(I)*0.00062136362
C         CONVERT RATES TO MPH
      PGR(I)=PGR(I)*0.0372818
      CWR(I)=CWR(I)*0.0372818
  790 CONTINUE
      NM12=NM1+NM1
      PLTX(NM12+1)=0.0
      PLTX(NM12+2)=TINCR*FLOAT(NM1)*0.2
C
      PLMAX=0.0
      PLMIN=1.0E20
      DO 826 I=1,NM1
      PLTY(I)=PCNT(I)
      CHAR(I)=AT
      PLTY(I+NM1)=PCNR(I)
      CHAR(I+NM1)=AST
      PLMAX=AMAX1(PLMAX,PCNT(I),PCNR(I))
      PLMIN=AMIN1(PLMIN,PCNT(I),PCNR(I))
  826 CONTINUE
      PLMIN=FLOAT(IFIX(PLMIN))
      NDIG=IFIX(ALOG10(PLMAX))-2
      RNUM=10**IABS(NDIG)
      IF(NDIG.LT.0)RNUM=1.0/RNUM
      PLMAX=FLOAT(IFIX(PLMAX/RNUM+1))*RNUM
      PLTY(LAST+1)=PLMIN
      PLTY(LAST+2)=(PLMAX-PLMIN)*0.2
C
      CALL PLOTXY(PLTX,PLTY,NM1*2,CHAR,'TIME FROM START OF FIRE (MINUTES
     &)','PROBABILITY','PREDICTED SUCCESS (CONTAINMENT=@,CONTROL=*)')
  910 PRINT    2012
 2012 FORMAT(3X,43HDO YOU WISH TO SEE THE SUPPRESSION ANALYSIS,
     & 15H TABLE (YES/NO))
```

```
      PRINT    2009
      READ     2010,ANS
      IF(ANS.EQ.NO)GO TO 810
      IF(ANS.EQ.YES)GO TO 920
      PRINT    2011
      GO TO 910
  920 ITINCR=IFIX(TINCR+0.00001)
      PRINT    2013,ITINCR
 2013 FORMAT(3X,44HENTER TIME INTERVAL IN MINUTES (MULTIPLE OF ,
     & I2,9H MINUTES))
      PRINT    2009
      READ     2015,TINTER
 2015 FORMAT(V)
      IF(TINTER.LE.0.0001)TINTER=TINCR
      REM=AMOD(TINTER,TINCR)
      IF(REM.GT.0.00001)TINTER=TINTER+TINCR-REM
      INCR=IFIX((TINTER+0.000001)/TINCR)
      INCR=MIN0(5,INCR)
C
C
C     PRINT TIME HISTORY TABLE
C
  750 PRINT    2005
 2005 FORMAT(/20X,26HSUPPRESSION ANALYSIS TABLE/
     & 23X,51HPERIMETER             CONTROL LINE          PROBABILITY/
     & 5X,11HDATE   TIME,          14X,6HGROWTH,14X,6HGROWTH,11X,2HOF/
     & 20X,58HLENGTH      RATE      LENGTH      RATE    CONTAINMENT CONTROL
     &/20X,36H(MILES)     (MPH)     (MILES)     (MPH) /)
 2007 FORMAT(3X,I2,2(1H/,2I1),1X,2I1,1H:,2I1,4(2X,F8.3),5X,2H0.,2I1,6X,2H0.,
     & 2I1)
C         CONVERT UNITS AND OUTPUT EACH LINE OF TABLE
      DO 800 I=1,NM1,INCR
C         CONVERT TIME TO DATE AND TIME
      CALL OUTTIM(TOF,JDOF,IYRF,PLTX(I),MONC,JDAYC,IYRC,IHRS,IMINC)
C         TRUNCATE PROBABILITY OF CONTAINMENT AND CONTROL TO 2 DIGITS
      IPCNC=IFIX(PCNT(I)*100.0+0.5)
      IPCRC=IFIX(PCNR(I)*100.0+0.5)
      IQPND1=IPCNC/10
      IQPND2=IPCNC-IQPND1*10
      IQPRD1=IPCRC/10
      IQPRD2=IPCRC-IQPRD1*10
      IQDD1=JDAYC/10
      IQDD2=JDAYC-IQDD1*10
      IQYD1=IYRC/10
      IQYD2=IYRC-IQYD1*10
      IQHD1=IHRS/10
      IQHD2=IHRS-IQHD1*10
      IQMD1=IMINC/10
      IQMD2=IMINC-IQMD1*10
C         OUTPUT
      PRINT    2007,MONC,IQDD1,IQDD2,IQYD1,IQYD2,IQHD1,IQHD2,
     & IQMD1,IQMD2,PL(I),PGR(I),WD(I),CWR(I),IQPND1,IQPND2,
     & IQPRD1,IQPRD2
  800 CONTINUE
      PRINT    2007
C
C     DETERMINE IF USER WANTS PLOTS
C
  810 PRINT    2008
 2008 FORMAT(3X,44HDO YOU WISH TO SEE THE DISTANCE PREDICTION P,
     & 12HLOT (YES/NO))
      PRINT    2009
 2009 FORMAT(7X,4H....)
      READ     2010,ANS
 2010 FORMAT(A4)
      IF(ANS.EQ.NO)GO TO 930
```

```
      IF(ANS.EQ.YES)GO TO 820
      PRINT    2011
 2011 FORMAT(3X,35H*** ERROR *** UNRECOGNIZABLE ANSWER /
     &        3X,34H                   INPUT ONLY YES OR NO)
      GO TO 810
C
C     PLOT RESULTS
C
  820 PLMAX=0.0
      PLMIN=1.0E20
      DO 822 I=1,NM1
      PLTY(I)=PL(I)
      PLTY(I+NM1)=WD(I)
      CHAR(I)=ZER
      CHAR(I+NM1)=PLUS
      PLMAX=AMAX1(PLMAX,PL(I),WD(I))
      PLMIN=AMIN1(PLMIN,PL(I),WD(I))
  822 CONTINUE
      PLMIN=FLOAT(IFIX(PLMIN))
      NDIG=IFIX(ALOG10(PLMAX))-2
      RNUM=10**IABS(NDIG)
      IF(NDIG.LT.0)RNUM=1.0/RNUM
      PLMAX=FLOAT(IFIX(PLMAX/RNUM+1.0))*RNUM
      LAST=2*NM1
      PLTY(LAST+1)=PLMIN
      PLTY(LAST+2)=(PLMAX-PLMIN)*0.2
C
      CALL PLOTXY(PLTX,PLTY,NM1*2,CHAR,'TIME FROM START OF FIRE (MINUTES
     &)','MILES','DIST. PREDICTIONS (PER. LENGTH=0,WORK DONE=+)'        )
  930 PRINT    2016
 2016 FORMAT(3X,19HDO YOU WISH TO SEE ,
     & 33HTHE RATE PREDICTION PLOT (YES/NO))
      PRINT    2009
      READ     2010,ANS
      IF(ANS.EQ.NO)GO TO 900
      IF(ANS.EQ.YES)GO TO 940
      PRINT    2011
      GO TO 930
C
  940 PLMAX=0.0
      PLMIN=1.0E20
      DO 824 I=1,NM1
      PLTY(I)=PGR(I)
      PLTY(I+NM1)=CWR(I)
      PLMAX=AMAX1(PLMAX,PGR(I),CWR(I))
      PLMIN=AMIN1(PLMIN,PGR(I),CWR(I))
  824 CONTINUE
      PLMIN=FLOAT(IFIX(PLMIN))
      NDIG=IFIX(ALOG10(PLMAX))-2
      RNUM=10**IABS(NDIG)
      IF(NDIG.LT.0)RNUM=1.0/RNUM
      PLMAX=FLOAT(IFIX(PLMAX/RNUM+1.0))*RNUM
      PLTY(LAST+1)=PLMIN
      PLTY(LAST+2)=(PLMAX-PLMIN)*0.2
C
      CALL PLOTXY(PLTX,PLTY,NM1*2,CHAR,'TIME FROM START OF FIRE (MINUTES
     &)','MILES PER HOUR','RATE PREDICTIONS (PER. GROWTH=0,WORK RATE=+)'
     &)
C
  900 RETURN
      END
```

44.0 AMOUNT OF WORK DONE (WRKDUN) Version 1.00 15 June 1975

1. PURPOSE

This routine computes the amount of work done by a resource over an arbitrary time interval for initial attack evaluation.

2. ARGUMENTS

INPUTS

IDEXRS	-	the resource index in Resource Characteristics List
TI	-	the start time of the work interval relative to the resource's ETA (min)
TF	-	the finish time of the work interval relative to the resource's ETA (min)
TCY	-	the air drop resource cycle time (min)
WIDTH	-	the line width (meters)
WRRES	-	the initial attack effective work rate array
NRES	-	the number of entries in WRRES

INPUT/OUTPUT

WATER	-	the water available to, or used by the resource (liters/resource) On input; WATER = on-site water available On output; WATER = on-site water used during time interval

OUTPUT

WRKDUN	-	the amount of line built by the resource in the time interval (meters)

3. PROCEDURE

The work accomplished is estimated using the work rates derived by WKRTIA under local conditions, and for a specified resource type, start time, end time, cycle time (for air drops), water available on site (for engines), and a specified line width. The start and finish times are determined, the work rate of the specified resource is modified as necessary by line width considerations and work rate is accumulated over the work interval to obtain the work accomplished.

A few minor complexities are involved and these are briefly described below:

- In initial attack, the length of time which an engine can work is very limited, since they are generally not accompanied by tankers and must rely on the small amount of water carried on board. For this reason the engine is allowed to work until its water supply (i.e., the water available on site) has been exhausted. After this time the engine crew is assumed to work as a hand crew.
- The on-site water, if any, used by an engine is accumulated over the time interval and this amount is returned as output, so that the on-site water may be continuously updated in the Calculate Initial Control Routine.
- Hand crew work rates are modified by two factors; time lost due to rest breaks, etc., and decreasing work rate as a function of time on-line due to fatigue.
- Air drop work rates are modified by two factors which define the number of additional drops necessary to satisfy specified line width, if the specified line width is greater than the air drop pattern width, and the number of additional drops required to provide suitable saturation of fuel with retardant.

4. MODEL RESULTS

Cumulative line production is presented for six classes (including sub-classes) of resources under three sets of conditions: (1) Moderate work and burning conditions, (2) severe work and burning conditions, and (3) very severe work and burning conditions. The characteristics employed to describe these conditions are shown below:

	Moderate	Severe	Very Severe	Work	Burn-ing
Fuel type	grass	chamise	chaparral	X	X
Fuel age	1 year	15 years	25 years	X	X
Day of year	∿15 June	∿1 Aug.	∿1 Oct.		X
Wind speed	10 mph	20 mph	40 mph	X	X
Slope	0%	30%	60%	X	X
Surface roughness	0 m	0.3 m	0.6 m	X	

Figures 63, 64 and 65 show cumulative line production plotted versus applied time for the three sets of work and burning conditions described above. It can be seen that cumulative line production increases monotonically with applied time, but that the rate of increase decreases with increasing applied time (due to fatigue) and that the rate of increase decreases also with the severity of the work conditions.

Figure 66 shows cumulative line production at 6.78 hours applied plotted for six resource classes, but omitting sub-classes. It can be seen that dozers and ground transported hand crews are the most effective resources under moderate conditions, but that air drops are the most effective under very severe conditions. Although only engines and air transported hand crews exhibit a null capability under some of the conditions of Figure 66, it should be realized that possible conditions may exist which produce a null capability in each of the six classes of resources. For air drops, turbulence and darkness, neither of which were considered here, are probably the controlling factors.

5.0 Flow Chart

See Figure 67.

Figure 63 Cumulative Resource Production Versus Applied Time for Moderate Work and Burning Conditions

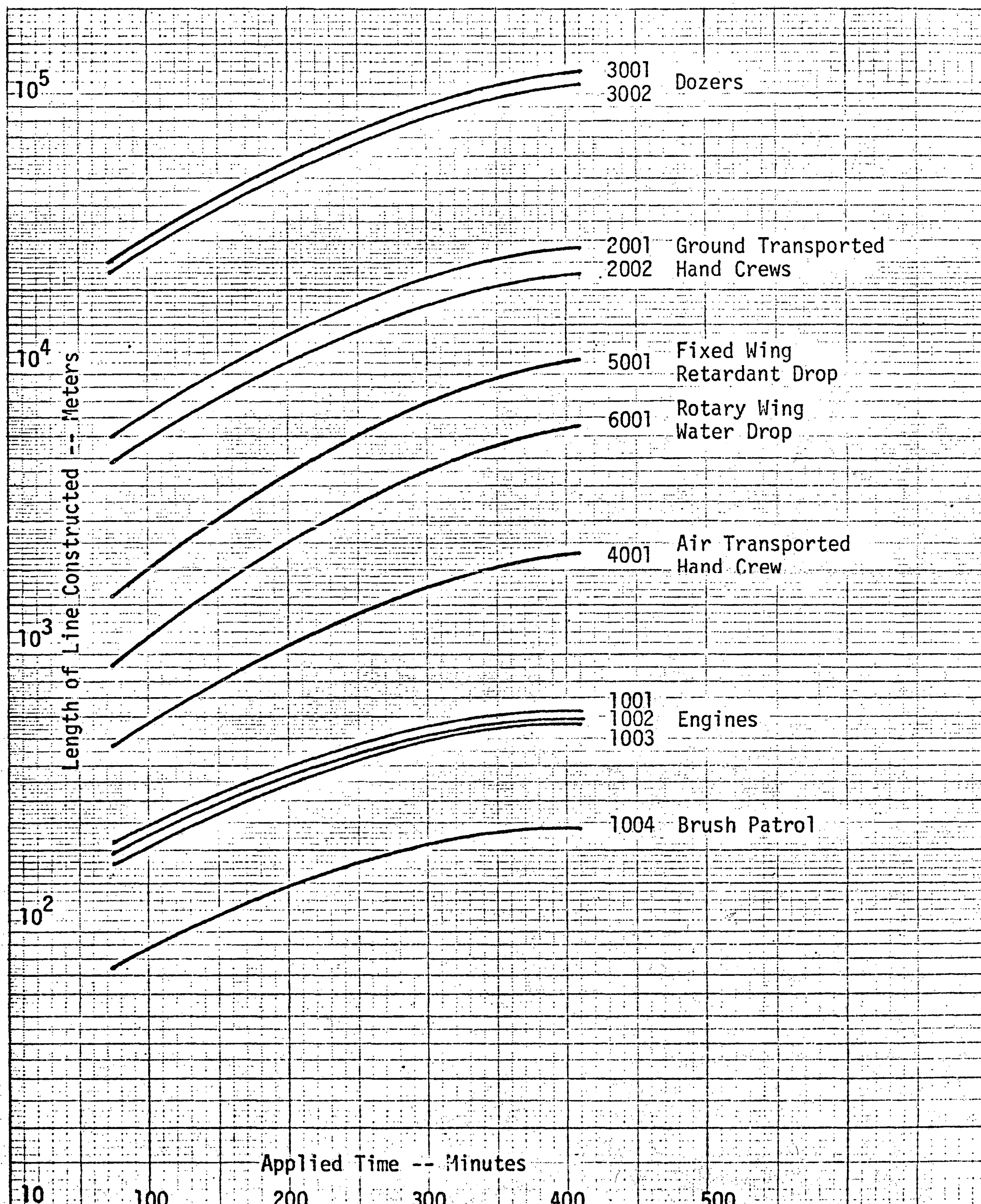


Figure 64 Cumulative Resource Production Versus Applied Time For Severe Work and Burning Conditions

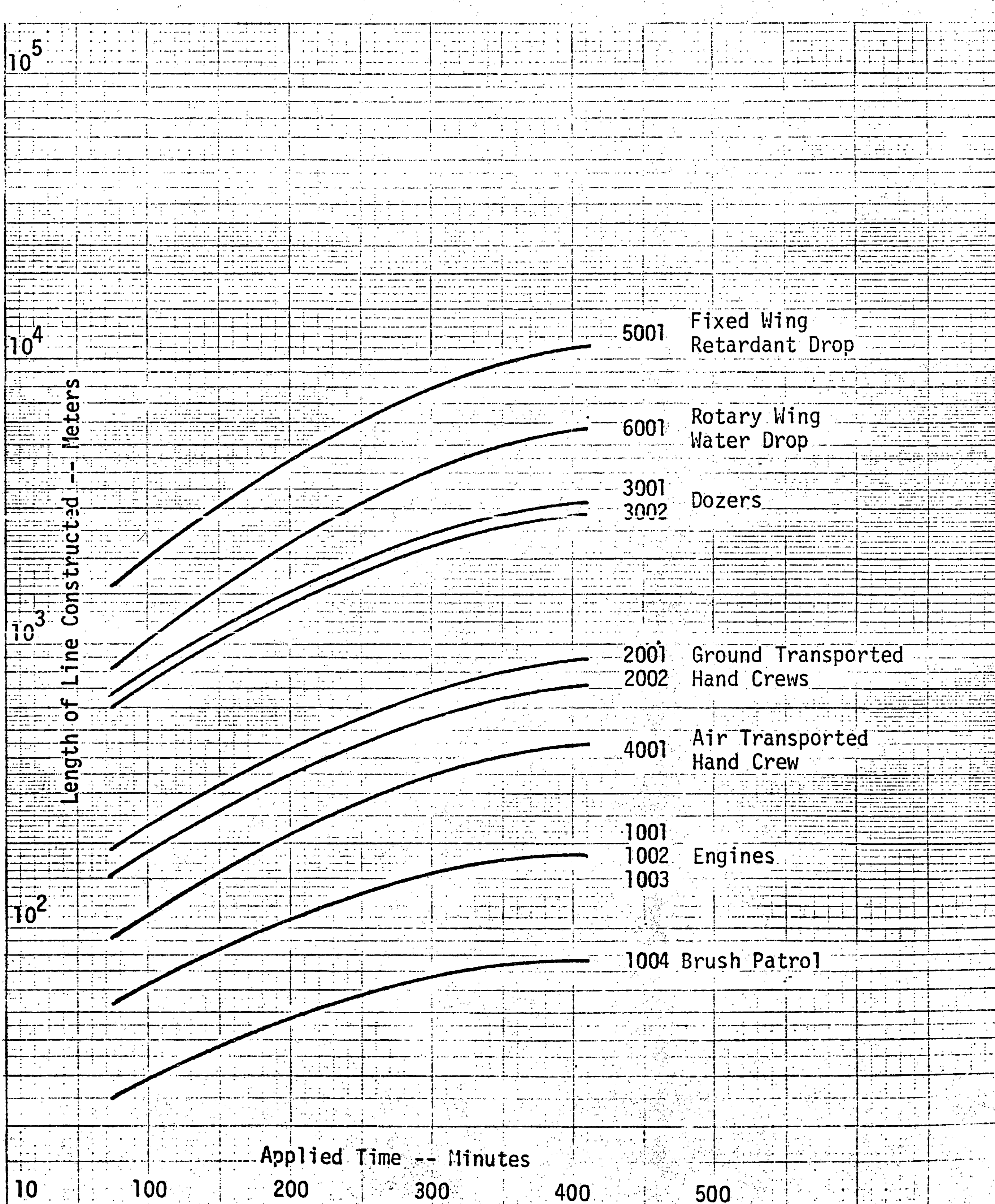


Figure 65 Cumulative Resource Production Versus Applied Time for Very Severe Work and Burning Conditions

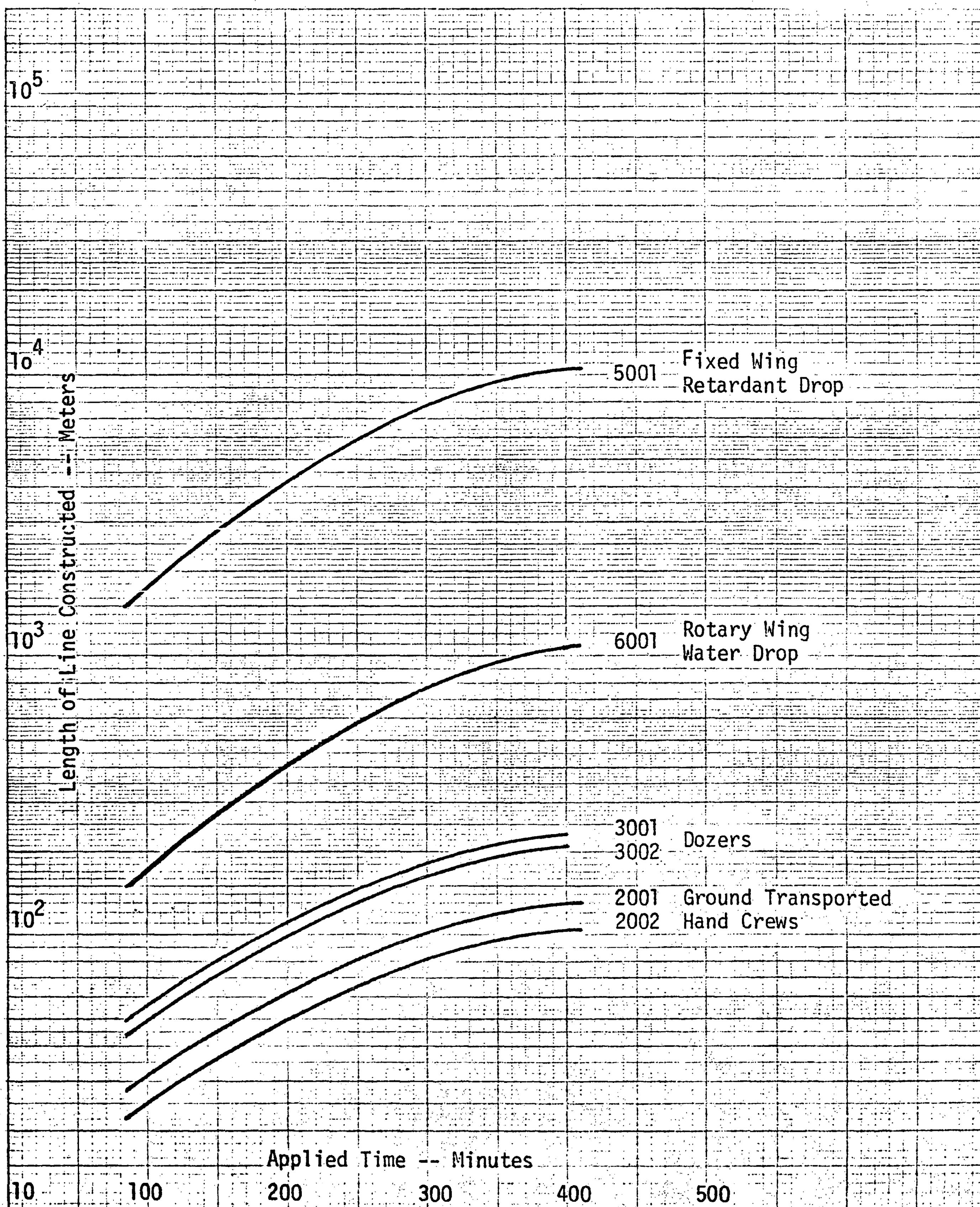


Figure 66 Cumulative Resource Production After 6.78 Hours Applied Time Versus Relative Severity of Work and Burning Conditions

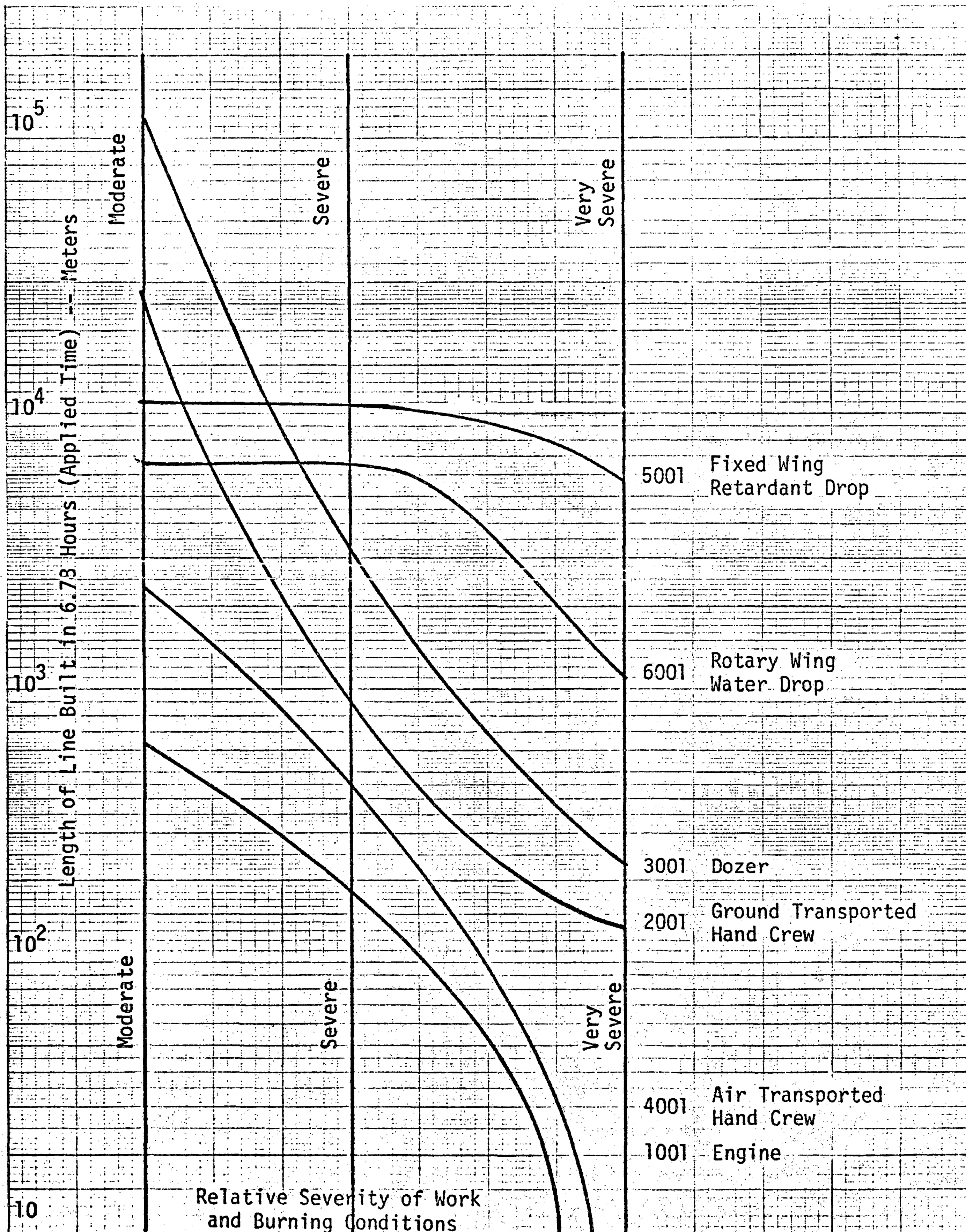


Figure 67   Amount of Work Done

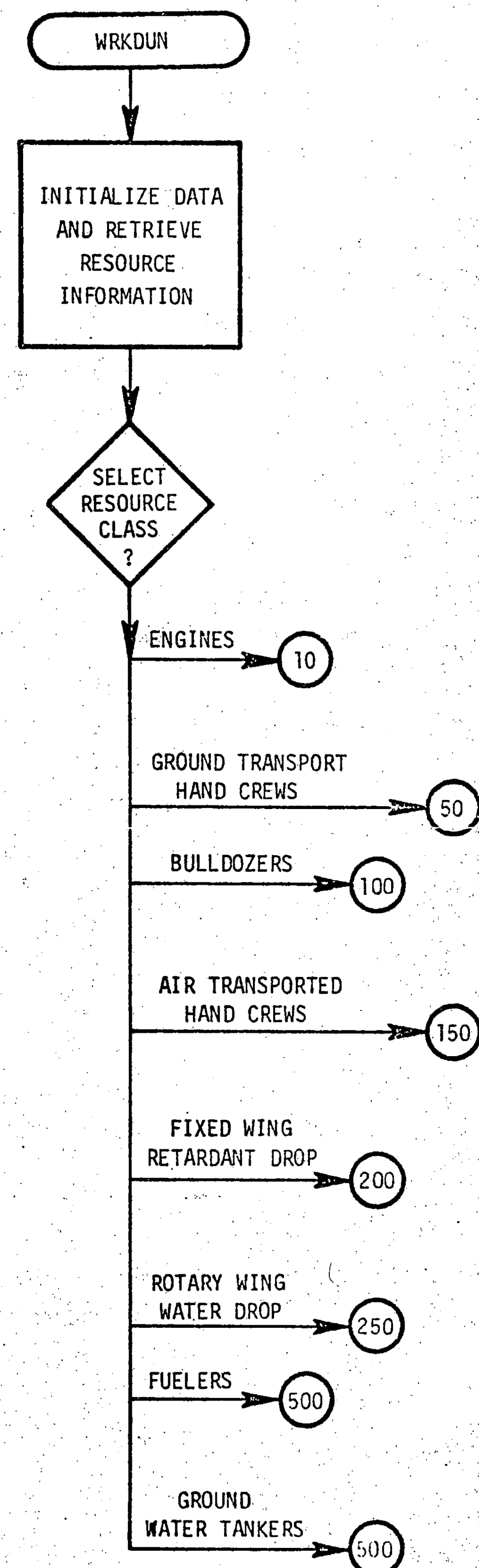


Figure 67 Amount of Work Done (Cont'd)

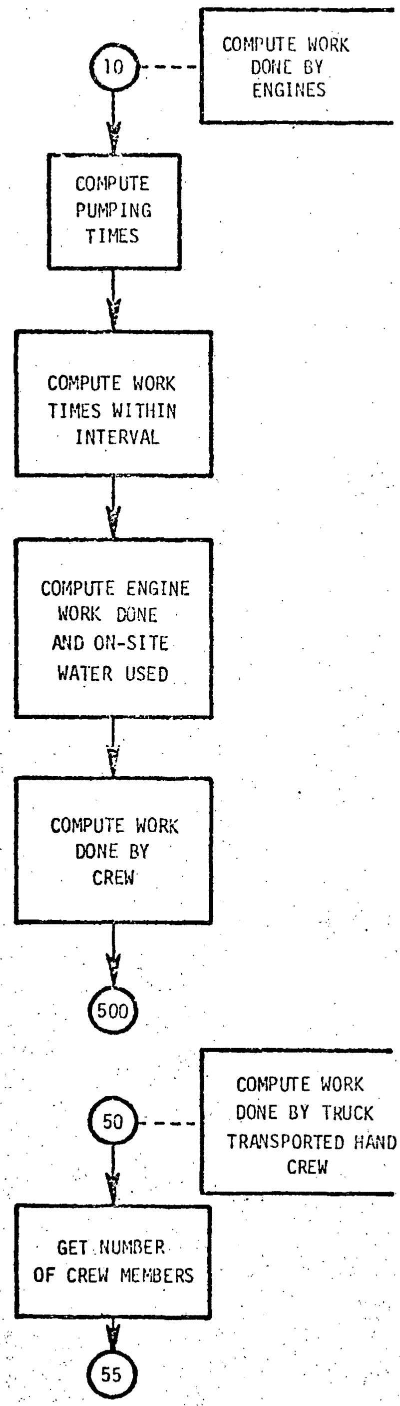


Figure 67 Amount of Work Done (Cont'd)

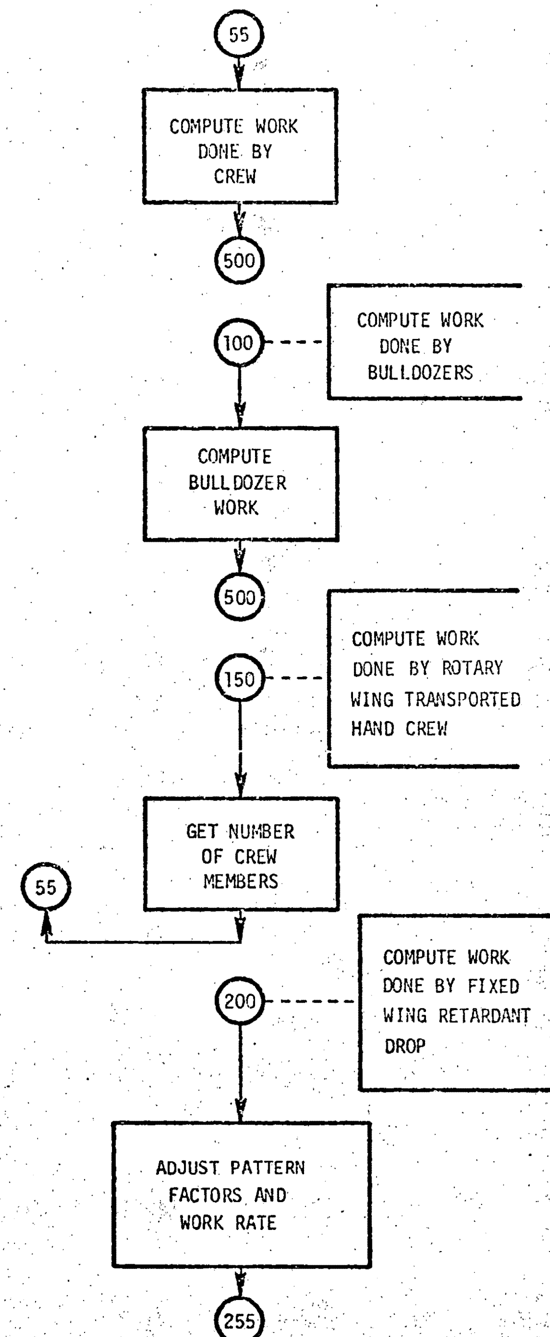
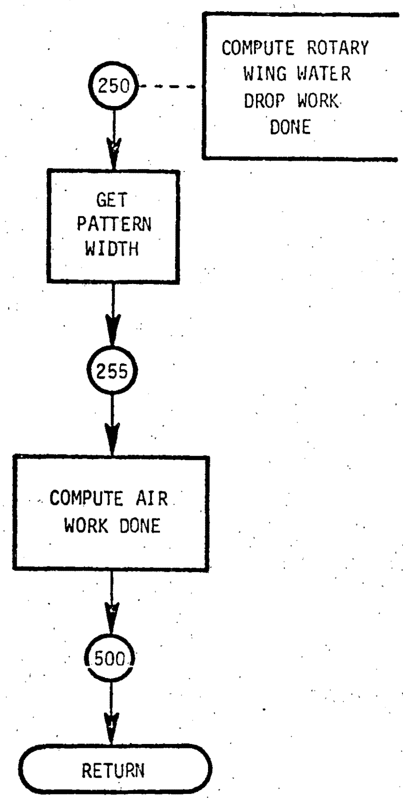


Figure 67 Amount of Work Done (Cont'd)



```
      FUNCTION WRKDUN(IDEXRS,TI,TF,TCY,WATER,WIDTH,WRRES,NRES)
C
C     THIS ROUTINE COMPUTES THE AMOUNT OF WORK DONE BY A
C     RESOURCE OVER AN ARBITRARY TIME INTERVAL
C     VERSION 1.00     2/11/76
C
C     PROGRAMMER - J. KEEFER
C
C     MODEL - J. SANDERLIN
C
C          INPUTS
C     IDEXRS - THE RESOURCE INDEX IN THE RESOURCE
C              CHARACTERISTICS LIST
C     TI     - THE INITIAL TIME OF THE WORK INTERVAL RELATIVE
C              TO THE RESOURCE ETA(MINUTES)
C     TF     - THE FINAL TIME OF THE WORK INTERVAL RELATIVE
C              TO THE RESOURCE ETA(MINUTES)
C     TCY    - AIR DROP RESOURCE CYCLE TIME IF NEEDED(MINUTES)
C     WIDTH  - THE LINE WIDTH(METER)
C     WRRES  - THE INITIAL ATTACK MAXIMUM EFFECTIVE WORK
C              RATE ARRAY; SEE SUBROUTINE WKRTIA FOR UNITS
C          INPUT/OUTPUT
C     WATER  - ON INPUT, THE ON-SITE WATER AVAILABLE PER RESOURCE
C              ON OUTPUT, THE AMOUNT OF ON-SITE WATER USED PER
C              RESOURCE(LITER)
C          OUTPUT
C     WRKDUN - THE LENGTH OF LINE BUILT BY THE GIVEN RESOURCE
C               FOR THE TIME INTERVAL SPECIFIED
C
C          ROUTINES USED
C     WRFATA,WRFAT,DWKTIM
C
      COMMON/RSDATA/ DUM(19),RESCHR(15,23),DZBL(5),ADWF(2)
C
      DIMENSION WRRES(NRES),IRESCH(15,23)
C
      EQUIVALENCE(IRESCH(1,1),RESCHR(1,1))
C
C
C     INITIALIZE DATA AND RETRIEVE RESOURCE INFORMATION
C
      WRKDUN = 0.0
      WRATE = WRRES(IDEXRS)
      TDEL = TF - TI
      ID = IRESCH(IDEXRS,3)
      ICLA = ID/1000
      ITYPE = ID - ICLA*1000
      GO TO(10,50,100,150,200,250,500,500), ICLA
C
C     ENGINE WORK DONE HERE
C          COMPUTE ENGINE PUMPING TIMES FOR ON BOARD AND/OR
C          ON-SITE WATER
C
   10 CONTINUE
      HOSCAP = RESCHR(IDEXRS,16)
      TENG1 = RESCHR(IDEXRS,12)/HOSCAP
      TENG2 = WATER/HOSCAP
C
C     COMPUTE WORK TIMES WITHIN INTERVAL FOR ENGINE
C          ON BOARD WATER, ON-SITE WATER AND CREW
C
      DTE1 = AMAX1(0.0,AMIN1(TDEL,TENG1-TI))
      DTE2 = AMAX1(0.0,AMIN1(TDEL-DTE1,TENG2))
      DTE = DTE1 + DTE2
      DTHC = TDEL - DTE
```

```
C
C     COMPUTE ON-SITE WATER USED PER ENGINE
C          AND ENGINE WORK DONE
C
      WATER = DTE2*HOSCAP
      IF(DTE.GT.0.0) WRKDUN = WRATE*DTE
      IF(WRATE .LE. 0.0)DTHC=DTE
      IF(DTHC.EQ.0.0) GO TO 500
C
C     COMPUTE WORK DONE BY ENGINE CREW
C          INCLUDING FATIGUE EFFECTS
C
      WRATE = WRRES(5)
      CREW = FLOAT(IRESCH(IDEXRS,21))
      TOL = TI + DTE
      FATWR = WRFATA(WRATE,TOL,DTHC)
      WRKDUN = WRKDUN + FATWR*CREW*DWKTIM(DTHC,TOL)/WIDTH
      GO TO 500
C
C     TRUCK TRANSPORTED HAND CREW WORK DONE HERE
C          RETRIEVE CREW NUMBER
C
   50 CONTINUE
      CREW = FLOAT(IRESCH(IDEXRS,21))
C
C          COMPUTE WORK DONE INCLUDING THE EFFECTS OF FATIGUE
C
   55 CONTINUE
      FATWR = WRFAT(WRATE,TI)
      WRKDUN = CREW*FATWR*DWKTIM(TDEL,TI)/WIDTH
      GO TO 500
C
C     BULLDOZER WORK DONE
C
  100 CONTINUE
      PASSES = FLOAT(1 + IFIX(WIDTH/DZBL(ITYPE)))
      WRKDUN = WRATE*TDEL/PASSES
      GO TO 500
C
C     ROTARY WING TRANSPORTED HAND CREW WORK DONE HERE
C          GET PERSONNEL NUMBER
C
  150 CONTINUE
      CREW = FLOAT(IRESCH(IDEXRS,21))
      GO TO 55
C
C     FIXED WING RETARDENT DROP WORK DONE HERE
C          GET PATTERN WIDTH FACTOR AND ADJUST PATTERN LENGTH
C
  200 CONTINUE
      PWIDF = ADWF(1)
      CAP = RESCHR(IDEXRS,12)
      WRATE = WRATE + 0.3*(CAP - 700.0*3.785)
      GO TO 255
C
C     ROTARY WING WATER DROP WORK DONE HERE
C          GET THE PATTERN WIDTH FACTOR
C
  250 CONTINUE
      PWIDF = ADWF(2)
C
C     COMPUTE AIR DROP WORK
C
  255 CONTINUE
      PWIDF = PWIDF*WRATE
      IF(PWIDF.LE.0.0001)PWIDF=0.000001*WIDTH
      PASSES = AMAX0(1,IFIX(WIDTH/PWIDF))
      DROPS = FLOAT(1 + IFIX(TDEL/TCY))
      WRKDUN = WRATE*DROPS/PASSES
  500 CONTINUE
      RETURN
      END
```

45.0 INITIAL ATTACK WORK RATES (WKRTIA) Version 1.00 15 June 1975

1. PURPOSE

This subroutine computes initial attack work rates for a specified set of conditions.

2. ARGUMENTS

INPUTS

FLOD	-	fuel loading (Kg/meter^2)
WNDS	-	wind speed (meter/min)
SLPE	-	slope (percent)
SRUF	-	surface roughness measure (meter)
TEMP	-	temperature (°C)
DISTOR	-	off-road distance (meter)

OUTPUTS

WRRES - effective work rate array

ENG,DOZ	-	in meters/min
HCT,RWHC	-	in meters^2/min/man
FWRD,RWWD	-	in meters/drop

3. PROCEDURE

The work rate modifiers for each resource type were found as a function of significant values of each factor that affects work rates. These modifiers were obtained from the detailed work rate algorithms for values of the affecting factors which cause significant changes in work rate. Ordered tables were formed (monotonic increasing) of the values of the affecting factors and the results are shown in Table 6. Tables were also formed of the values of the work rate modifiers for each resource type as a function of the value affecting the work rate. These results are displayed in Tables 7 through 12.

Table 6. Values of Factors Affecting Work Rates

Entry	FACTOR					
	Fuel Loading (tons/acre)	Wind Speed (mph)	Slope (%)	Surface Roughness (feet)	Temperature (°F)	Off-Road Distance (miles)
1	1.0	20.0	20.0	1.4	120.0	0.2
2	2.5	25.0	22.0	2.5	140.0	0.5
3	3.0	35.0	35.0	3.4	160.0	1.0
4	4.0	40.0	40.0	3.5	165.0	2.0
5	5.0	45.0	45.0	4.0		
6	6.0	50.0	50.0	4.5		
7	7.0	65.0	60.0	5.2		
8	8.0	70.0	70.0			
9	10.0	80.0				
10	11.0	95.0				
11	15.0					
12	17.0					

Table 7. Work Rate Modifiers for Engines

Entry	FACTOR					
	Fuel Loading (tons/acre)	Wind Speed (mph)	Slope (%)	Surface Roughness (feet)	Temper-ature (°F)	Off-Road Distance (miles)
1	-0.2	0.0	0.0	0.0	-0.2	-1.0
2	-0.2	-0.2	0.0	0.0	-0.5	-1.0
3	-0.2	-0.5	0.0	0.0	-0.8	-1.0
4	-0.5	-0.5	0.0	0.0	-1.0	-1.0
5	-0.5	-0.8	0.0	0.0		
6	-0.5	-1.0	0.0	0.0		
7	-0.5	-1.0	0.0	0.0		
8	-0.5	-1.0	0.0			
9	-0.5	-1.0				
10	-1.0	-1.0				
11	-1.0					
12	-1.0					

Table 8. Work Rate Modifiers for Truck Transported Hand Crews

Entry	FACTOR					
	Fuel Loading (tons/acre)	Wind Speed (mph)	Slope (%)	Surface Roughness (feet)	Temperature (°F)	Off-Road Distance (miles)
1	0.0	0.0	-0.2	0.0	-0.2	-0.2
2	0.0	0.0	-0.2	-0.2	-0.5	-0.5
3	0.0	0.0	-0.2	-0.2	-0.8	-0.8
4	0.0	-0.2	-0.5	-0.5	-1.0	-1.0
5	-0.2	-0.2	-0.5	-0.5		
6	-0.2	-0.2	-0.5	-0.8		
7	-0.2	-0.5	-0.8	-1.0		
8	-0.2	-0.5	-1.0			
9	-0.5	-0.8				
10	-0.5	-1.0				
11	-0.8					
12	-1.0					

Table 9. Work Rate Modifiers for Bulldozers

Entry	FACTOR					
	Fuel Loading (tons/acre)	Wind Speed (mph)	Slope (%)	Surface Roughness (feet)	Temperature (°F)	Off-Road Distance (miles)
1	0.0	0.0	0.0	-0.2	-0.2	0.0
2	-0.2	0.0	-0.2	-0.5	-0.5	0.0
3	-0.2	0.0	-0.5	-0.8	-0.8	0.0
4	-0.2	-0.2	-0.5	-0.8	-1.0	0.0
5	-0.2	-0.2	-0.8	-1.0		
6	-0.2	-0.2	-1.0	-1.0		
7	-0.5	-0.5	-1.0	-1.0		
8	-0.8	-0.5	-1.0			
9	-0.8	-0.8				
10	-0.8	-1.0				
11	-0.8					
12	-0.8					

Table 10. Work Rate Modifiers for Air Transported Hand Crews

Entry	FACTOR					
	Fuel Loading (tons/acre)	Wind Speed (mph)	Slope (%)	Surface Roughness (feet)	Temperature (°F)	Off-Road Distance (miles)
1	0.0	-0.2	-0.2	0.0	-0.2	0.0
2	0.0	-0.2	-0.2	-0.2	-0.5	0.0
3	0.0	-0.5	-0.2	-0.2	-0.8	0.0
4	0.0	-0.5	-0.5	-0.5	-1.0	0.0
5	-0.2	-0.8	-0.5	-0.5		
6	-0.2	-1.0	-0.5	-0.8		
7	-0.2	-1.0	-0.8	-1.0		
8	-0.2	-1.0	-1.0			
9	-0.5	-1.0				
10	-0.5	-1.0				
11	-0.8					
12	-1.0					

Table 11. Work Rate Modifiers for Fixed Wing Retardant Drop Aircraft

Entry	FACTOR					
	Fuel Loading (tons/acre)	Wind Speed (mph)	Slope (%)	Surface Roughness (feet)	Temper-ature (°F)	Off-Road Distance (miles)
1	-0.2	0.0	0.0	0.0	0.0	0.0
2	-0.2	-0.2	0.0	0.0	0.0	0.0
3	-0.5	-0.2	0.0	0.0	0.0	0.0
4	-0.5	-0.2	0.0	0.0	0.0	0.0
5	-0.5	-0.2	0.0	0.0		
6	-0.5	-0.5	0.0	0.0		
7	-0.8	-0.5	0.0	0.0		
8	-0.8	-0.8	0.0			
9	-0.8	-1.0				
10	-0.8	-1.0				
11	-1.0					
12	-1.0					

Table 12. Work Rate Modifiers for Rotary Wing Water Drop Aircraft

Entry	FACTOR					
	Fuel Loading (tons/acre)	Wind Speed (mph)	Slope (%)	Surface Roughness (feet)	Temper-ature (°F)	Off-Road Distance (miles)
1	0.0	-0.2	0.0	0.0	0.0	0.0
2	0.0	-0.2	0.0	0.0	0.0	0.0
3	0.0	-0.5	0.0	0.0	0.0	0.0
4	-0.2	-0.5	0.0	0.0	0.0	0.0
5	-0.2	-0.8	0.0	0.0		
6	-0.2	-1.0	0.0	0.0		
7	-0.2	-1.0	0.0	0.0		
8	-0.5	-1.0	0.0			
9	-0.8	-1.0				
10	-0.8	-1.0				
11	-1.0					
12	-1.0					

Given the values of the factors affecting work rate, which define the local conditions under which work is to be performed, these tables can be used to define an array of work rate modifiers for each resource type as a function of the specified values of the affecting factors. Upon summing all of the modifiers for each resource type, a linear combination defining the total work rate modifier for each resource type under the specified conditions is obtained. The product of the total work rate modifier with the maximum work rate for each respective resource yields the estimated work rates of each resource under the specified conditions.

4. COMMENTS

The procedure used here is a modification of the algorithm described in Reference 1 (MRC-7416-2-874, August 1974). The table look-up format is rapid and easily up-dated as new data are obtained.

5. FLOW CHART

See Figure 68.

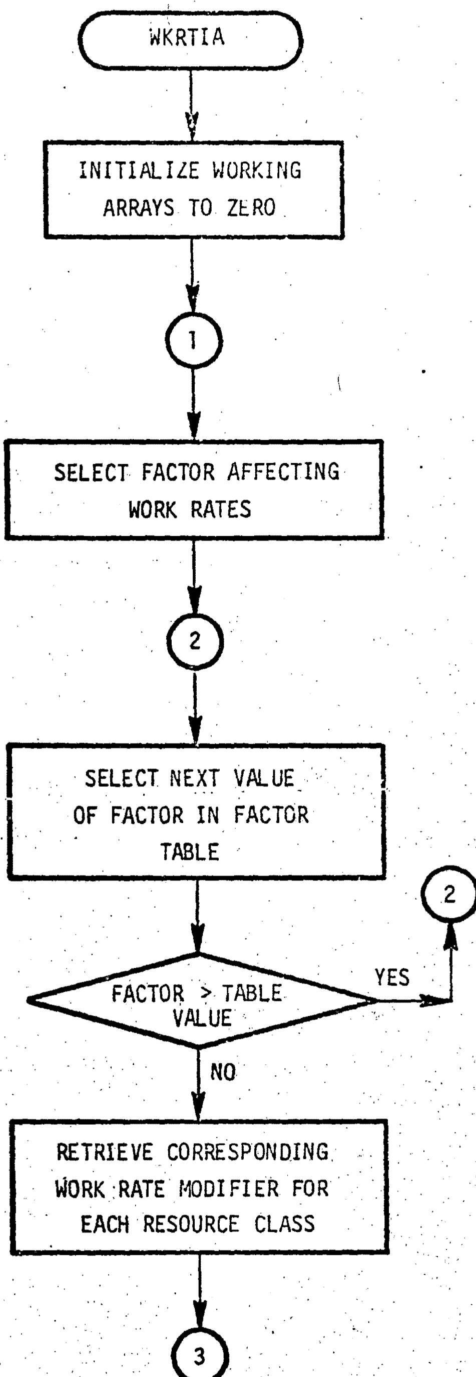


Figure 68 Logic Structure for WKRTIA

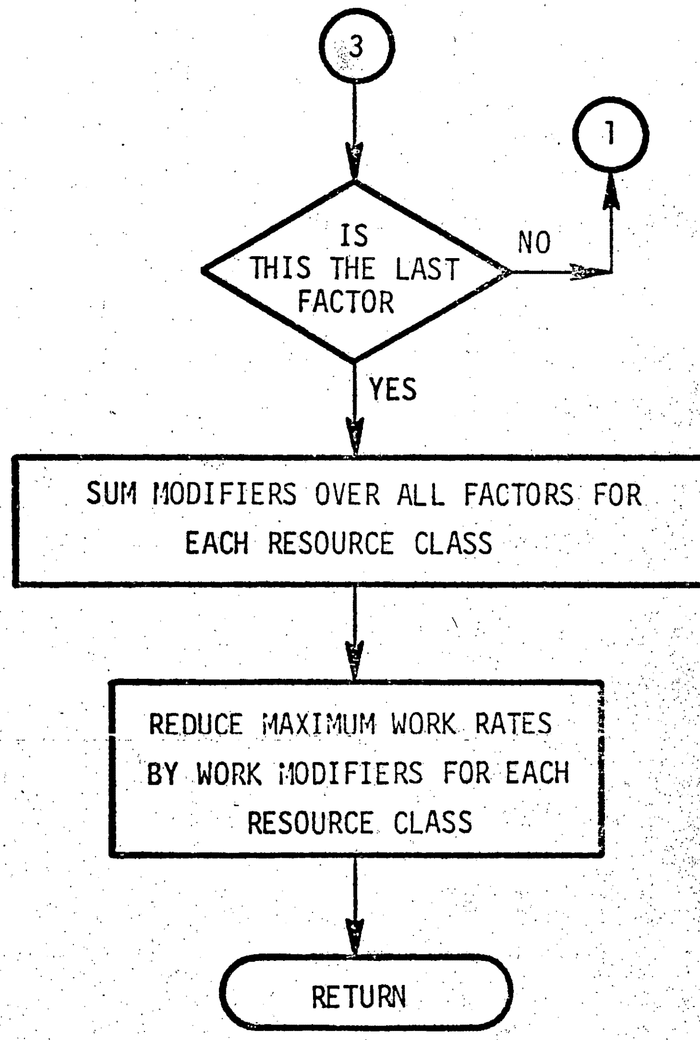


Figure 68 Logic Structure for WKRTIA (Cont'd)

```
      SUBROUTINE WKRTIA(FLOD,WNDS,SLPE,SRUF,TEMP,DISTOR,WRRES)
C
C     THIS SUBROUTINE COMPUTES INITIAL ATTACK WORK RATES FOR A
C     SPECIFIED SET OF ENVIRONMENTAL CONDITIONS
C     VERSION 1.00      2/11/76
C
C     PROGRAMMER - J. KEEFER
C
C     MODEL - J. SANDERLIN
C
C         INPUTS
C     FLOD   - THE FUEL LOADING(KG/METER SQUARED)
C     WNDS   - THE WIND SPEED(METER/MINUTE)
C     SLPE   - SLOPE(PERCENT)
C     SRUF   - THE SURFACE ROUGHNESS MEASURE(METER)
C     TEMP   - THE TEMPERATURE(DEG. C)
C     DISTOR - THE OFF-ROAD DISTANCE TO THE FIRE(METER)
C         OUTPUTS
C     WRRES  - THE MAXIMUM WORK RATE ARRAY FOR THE INPUT
C              CONDITIONS
C              ENG,DOZ  - METER/MINUTES
C              HCT,RWHC  - METER-SQ./MINUTE/MAN
C              FWAD,RWWD - METER/DROP
C
      COMMON /RSDATA/ DUM1(16),MAXR,MAXS,MAXC,RESCHR(15,23),AUX(8)
C
      DIMENSION IRESCH(15,23)
C
      DIMENSION VAL(6),WR(6,6),WRFAC(6),WRRES(15),NTAB(6)
      DIMENSION VALTAB(56),
     &     FLDRNG(12),WNDRNG(12),SLPRNG(10),
     &     SRFRNG(9),TMPRNG(7),ORDRNG(6)
      DIMENSION WRFTAB(56,6)
      DIMENSION ENGTAB(56),
     &     FLET(12),WSET(12),SLET(10),
     &     SRET(9),TPET(7),ORET(6)
      DIMENSION THCTAB(56),
     &     FLTT(12),WSTT(12),SLTT(10),
     &SRTT(9),TPTT(7),ORTT(6)
      DIMENSION DOZTAB(56),
     &     FLDT(12),WSDT(12),SLDT(10),
     &     SRDT(9),TPDT(7),ORDT(6)
      DIMENSION AHCTAB(56),
     &     FLAT(12),WSAT(12),SLAT(10),
     &     SRAT(9),TPAT(7),ORAT(6)
      DIMENSION FWATAB(56),
     &     FLFT(12),WSFT(12),SLFT(10),
     &     SRFT(9),TPFT(7),ORFT(6)
      DIMENSION RWATAB(56),
     &     FLRT(12),WSRT(12),SLRT(10),
     &     SRRT(9),TPRT(7),ORRT(6)
C
C     VALTAB IS MADE UP OF FLDRNG,WNDRNG,SLPRNG,
C     SRFRNG,TMPRNG,AND ORDRNG
C
      EQUIVALENCE(IRESCH(1,1),RESCHR(1,1))
C
```

```
      EQUIVALENCE(VALTAB(1),FLDRNG(1)),
     &           (VALTAB(13),WNDRNG(1)),
     &           (VALTAB(25),SLPRNG(1)),
     &           (VALTAB(35),SRFRNG(1)),
     &           (VALTAB(44),TMPRNG(1)),
     &           (VALTAB(51),ORDRNG(1))
C
C     WRFTAB IS MADE UP OF ENGTAB,THCTAB,
C     DOZTAB,AHCTAB,FWATAB, AND RWATAB
C
      EQUIVALENCE(WRFTAB(1,1),ENGTAB(1)),
     &           (WRFTAB(1,2),THCTAB(1)),
     &           (WRFTAB(1,3),DOZTAB(1)),
     &           (WRFTAB(1,4),AHCTAB(1)),
     &           (WRFTAB(1,5),FWATAB(1)),
     &           (WRFTAB(1,6),RWATAB(1))
C
C     ENGTAB IS MADE UP OF FLET,WSET
C     SLET,SRET,TPET, AND ORET
C
      EQUIVALENCE(ENGTAB(1),FLET(1)),
     &           (ENGTAB(13),WSET(1)),
     &           (ENGTAB(25),SLET(1)),
     &           (ENGTAB(35),SRET(1)),
     &           (ENGTAB(44),TPET(1)),
     &           (ENGTAB(51),ORET(1))
C
C     THCTAB IS MADE UP OF FLTT, WSTT,
C     SLTT, SRTT, TPTT, AND ORTT
C
      EQUIVALENCE(THCTAB(1),FLTT(1)),
     &           (THCTAB(13),WSTT(1)),
     &           (THCTAB(25),SLTT(1)),
     &           (THCTAB(35),SRTT(1)),
     &           (THCTAB(44),TPTT(1)),
     &           (THCTAB(51),ORTT(1))
C
C     DOZTAB IS MADE UP OF FLDE, WSDT,
C     SLDT, SRDT, TPDT, AND ORDT
C
      EQUIVALENCE(DOZTAB(1),FLDT(1)),
     &           (DOZTAB(13),WSDT(1)),
     &           (DOZTAB(25),SLDT(1)),
     &           (DOZTAB(35),SRDT(1)),
     &           (DOZTAB(44),TPDT(1)),
     &           (DOZTAB(51),ORDT(1))
C
C     AHCTAB IS MADE UP OF FLAT, WSAT,
C     SLAT, SRAT, TPAT, AND ORAT
C
      EQUIVALENCE(AHCTAB(1),FLAT(1)),
     &           (AHCTAB(13),WSAT(1)),
     &           (AHCTAB(25),SLAT(1)),
     &           (AHCTAB(35),SRAT(1)),
     &           (AHCTAB(44),TPAT(1)),
     &           (AHCTAB(51),ORAT(1))
C
C     FWATAB IS MADE UP OF FLFT, WSFT,
C     SLFT,SRFT,TPFT, AND ORFT
C
      EQUIVALENCE(FWATAB(1),FLFT(1)),
     &           (FWATAB(13),WSFT(1)),
     &           (FWATAB(25),SLFT(1)),
     &           (FWATAB(35),SRFT(1)),
     &           (FWATAB(44),TPFT(1)),
     &           (FWATAB(51),ORFT(1))
```

```
C
C     RWATAB IS MADE UP OF FLRT, WSRT,
C     SLRT, SRRT, TPRT, AND ORRT
C
      EQUIVALENCE(RWATAB(1),FLRT(1)),
     &           (RWATAB(13),WSRT(1)),
     &           (RWATAB(25),SLRT(1)),
     &           (RWATAB(35),SRRT(1)),
     &           (RWATAB(44),TPRT(1)),
     &           (RWATAB(51),ORRT(1))
C
C     VALTAB CONTAINS THE RANGE OF VALUES
C     OF THE ENVIRONMENTAL VARIABLES THAT
C     MAY AFFECT WORK RATES
C
C     DEFINE THE RANGE OF VALUES FOR
C     FUEL LOADS (KG/M**2)
C
      DATA FLDRNG/
     &     0.224,0.561,1.22,2.50,
     &     3.12,4.35,5.57,6.79,
     &     7.24,8.47,9.37,10.81/
C
C     DEFINE THE RANGE OF VALUES
C     FOR WIND SPEEDS (M/MIN)
C
      DATA WNDRNG/
     &     0000.0, 268.4, 563.4,670.5,
     &      938.7,1073.0,1207.0,1341.0,
     &     1743.0,1877.0,2146.0,2414.0/
C
C     DEFINE THE RANGE OF VALUES
C     FOR SLOPES (PERCENT)
C
      DATA SLPRNG/
     &     0.00,0.1,0.2,0.22,
     &     0.35,0.40,0.45,0.50,
     &     0.60,0.70/
C
C     DEFINE THE RANGE OF VALUES
C     FOR SURFACE ROUGHNESS (M)
C
      DATA SRFRNG/
     &     0.000,0.213,0.427,0.762,
     &     1.036,1.067,1.219,1.372,
     &     1.587/
C
C     DEFINE THE RANGE OF VALUES
C     FOR TEMPERATURE (DEG. C)
C
      DATA TMPRNG/
     &     10.0,20.0,30.0,48.9,
     &     60.0,71.1,73.9/
C
C     DEFINE THE RANGE OF VALUES
C     FOR OFF-ROAD DISTANCE (M)
C
      DATA ORDRNG/
     &     0000.0, 161.0, 321.8, 804.5,
     &     1609.0,3218.0/
C
C     WRFTAB CONTAINS FACTORS THAT AFFECT
C     WORK RATE. THESE FACTORS CORRESPOND
C     TO THE VALUES OF THE ENVIRONMENTAL
C     VARIABLES DEFINED IN VALTAB.
C     AND ARE  DIFFERENT FOR THE  SIX
```

```
C      CLASSES OF RESOURCES=
C            ENGINES, TRUCK TRANSPORTED
C            HAND CREWS, BULLDOZERS, AIR
C            TRANSPORTED HAND CREWS, FIXED WING
C            WING RETARDANT DROPS, AND ROTARY
C            WING WATER DROPS.
C
C      WORK RATE FACTORS FOR ENGINES
C      CORRESPONDING TO SPECIFIED (VALTAB)
C      VALUES OF=
C            FUEL LOADING
       DATA FLET/3*-0.2,6*-.4,3*-.80/
C            WINDSPEED
       DATA WSET/3*0.0,-0.2,2*-0.3,
     &           -0.5,5*-0.8/
C            SLOPE
       DATA SLET/10*0.0/
C            SURFACE ROUGHNESS
       DATA SRET/9*0.0/
C            TEMPERATURE
       DATA TPET/2*0.0,-0.1,-0.2,-0.5,
     &           -0.8,-1.0/
C            OFF-ROAD DISTANCE
       DATA ORET/0.0,-0.2,4*-1.0/
C
C      WORK RATE FACTORS FOR TRUCK TRANSPORTED
C      HAND CREWS CORRESPONDING TO SPECIFIED
C      VALUES (VALTAB) OF=
C            FUEL LOADING
       DATA FLIT/4*0.0,4*-0.2,2*-0.4,
     &           -0.6,-0.8/
C            WIND SPEED
       DATA WSIT/5*0.0,3*-0.2,2*-0.5,
     &           -0.8,-1.0/
C            SLOPE
       DATA SLIT/0.0,-0.1,3*-0.2,3*-0.3,
     &           -0.5,-0.8/
C            SURFACE ROUGHNESS
       DATA SRIT/3*0.0,2*-0.2,2*-0.5,
     &           -0.8,-1.0/
C            TEMPERATURE
       DATA TPIT/2*0.0,-0.1,-0.2,-0.5,
     &           -0.8,-1.0/
C            OFF-ROAD DISTANCE
       DATA ORIT/0.0,-0.1,-0.2,-0.5,
     &           -0.8,-1.0/
C
C      WORK RATE FACTORS FOR BULLDOZERS
C      CORRESPONDING TO SPECIFIED VALUES
C      (VALTAB) OF=
C            FUEL LOADING
       DATA FLDT/0.0,5*-0.2,-0.3,5*-0.4/
C            WIND SPEED
       DATA WSDT/5*0.0,3*-0.2,2*-0.3,
     &           -0.5,-1.0/
C            SLOPE
       DATA SLDT/3*0.0,-0.2,2*-0.3,
     &           -0.5,3*-0.8/
C            SURFACE ROUGHNESS
       DATA SRDT/0.0,-0.1,-0.2,-0.3,
     &           2*-0.5,3*-0.8/
C            TEMPERATURE
       DATA TPDT/2*0.0,-0.1,-0.2,-0.5,
     &           -0.8,-1.0/
C            OFF-ROAD DISTANCE
       DATA ORDT/6*0.0/
```

```
C
C     WORK RATE FACTORS FOR AIR TRANSPORTED
C     HAND CREWS CORRESPONDING TO SPECIFIED
C     (VALTAB) VALUES OF=
C           FUEL LOADING
      DATA FLAT/4*0.0,4*-0.2,2*-0.4,
     &            -0.6,-0.8/
C           WIND SPEED
      DATA WSAT/2*0.0,2*-0.2,2*-0.5,
     &            -0.8,5*-1.0/
C           SLOPE
      DATA SLAT/0.0,-0.1,3*-0.2,3*-0.3,
     &            -0.5,-0.8/
C           SURFACE ROUGHNESS
      DATA SRAT/3*0.0,2*-0.2,2*-0.5,
     &            -0.8,-1.0/
C           TEMPERATURE
      DATA TPAT/2*0.0,-0.1,-0.2,-0.5,
     &            -0.8,-1.0/
C           OFF-ROAD DISTANCE
      DATA ORAT/6*0.0/
C
C     WORK RATE FACTORS FOR FIXED WING RETARDANT
C     DROP AIRCRAFT CORRESPONDING TO
C     SPECIFIED (VALTAB) VALUES OF=
C           FUEL LOADING
      DATA FLFT/2*-0.2,3*-0.4,5*-0.5,
     &            2*-0.6/
C           WIND SPEED
      DATA WSFT/3*0.0,4*-0.2,2*-0.4,
     &            -0.6,2*-0.8/
C           SLOPE
      DATA SLFT/10*0.0/
C           SURFACE ROUGHNESS
      DATA SRFT/9*0.0/
C           TEMPERATURE
      DATA TPFT/7*0.0/
C           OFF-ROAD DISTANCE
      DATA ORFT/6*0.0/
C
C     WORK RATE FACTORS FOR ROTARY WING
C     WATER DROP AIRCRAFT CORRESPONDING
C     TO SPECIFIED (VALTAB) VALUES OF=
C           FUEL LOADING
      DATA FLRT/3*0.0,4*-0.2,-0.3,
     &            2*-0.5,2*-0.8/
C           WIND SPEED
      DATA WSRT/0.0,-0.1,2*-0.2,
     &            2*-0.5,-0.8,5*-1.0/
C           SLOPE
      DATA SLRT/10*0.0/
C           SURFACE ROUGHNESS
      DATA SRRT/9*0.0/
C           TEMPERATURE
      DATA TPRT/7*0.0/
C           OFF-ROAD DISTANCE
      DATA ORRT/6*0.0/
C
      DATA NTAB/12,12,10,9,7,6/, NFAC/6/
C
C     SET NUMBER OF CLASSES AND RESOURCES FOR WHICH WORK RATES
C     ARE TO BE COMPUTED. THE CODE IS SPECIFIC FOR THE PROTOTYPE
C     DATA BASE.
C
      NCLA = MAXC - 2
      NRES = MAXR - 4
```

```
C
C     FILL THE VALUE ARRAY
C
      VAL(1) = FLOD
      VAL(2) = WNDS
      VAL(3) = SLPE
      VAL(4) = SRUF
      VAL(5) = TEMP
      VAL(6) = DISTOR
C
C     INITIALIZE THE WORKING ARRAYS TO ZERO
C
      DO 20 J = 1,NCLA
      DO 10 I = 1,NFAC
      WR(J,I) = 0.0
   10 CONTINUE
      WRFAC(J) = 0.0
   20 CONTINUE
C
C     LOOP OVER FACTORS AFFECTING THE WORK RATES
C
      K1 = 1
      DO 60 I = 1,NFAC
      K2 = NTAB(I) + K1 -1
      VALUE = VAL(I)
C
C     SEARCH TABLE FOR APPROPRIATE ENTRY
C
      DO 30 K = K1,K2
      KDEX = K
      IF(VALUE.LE.VALTAB(K)) GO TO 40
   30 CONTINUE
C
C     TABLE ENTRY FOUND - STORE RESOURCE WORK RATE
C     REDUCTION FACTOR FOR EACH RESOURCE CLASS
C
   40 CONTINUE
      DO 50 J=1,NCLA
      WR2=WRFTAB(KDEX,J)
      IF(KDEX.EQ.K1)GO TO 45
      WR1=WRFTAB(KDEX-1,J)
      VAL1=VALTAB(KDEX-1)
      VAL2=VALTAB(KDEX)
      WR(J,I)=WR1+(WR2-WR1)*(VALUE-VAL1)/(VAL2-VAL1)
      GO TO 49
   45 WR(J,I)=WR2
   49 CONTINUE
   50 CONTINUE
      K1 = K2 + 1
   60 CONTINUE
C
C     SUM OVER REDUCTION FACTORS FOR EACH RESOURCE
C
      DO 80 J = 1,NCLA
      FAC = 0.0
      DO 70 I = 1,NFAC
      FAC = FAC + WR(J,I)
   70 CONTINUE
      WRFAC(J)  = 1.0 + AMAX1(FAC,-1.0)
   80 CONTINUE
C
C     REDUCE MAXIMUM WORK RATES BY REDUCTION FACTOR
C
      DO 100 I = 1,NRES
      ID = IRESCH(I,3)
      J = ID/1000
      IF(I .EQ. 3 .OR. I .EQ. 4)WRFAC(J)=WRFAC(J)
     &    -0.5*WR(J,6)
      WRRES(I) = RESCHR(I,23)*WRFAC(J)
  100 CONTINUE
C
      RETURN
      END
```

46.0 ACTUAL WORK TIME (WKTIM) Version 1.00 15 June 1975

1. PURPOSE

This routine calculates the time personnel actually work as a function of continuous on-line time.

2. ARGUMENTS

<u>INPUT</u>

OLTIM - time spent on-line by personnel (minutes)

<u>OUTPUTS</u>

WKTIM - time spent working on a line (minutes)

3. COMPUTATION PROCEDURE

The formula of Reference 12 is used to compute the actual work time, t_w, as a function of time spent on line, $t_{o\ell}$.

$$t_w = 0.96\, t_{o\ell}^{0.796},$$

where the times are in hours. If the time on-line is less than an hour, then t_w is taken to be equal to $t_{o\ell}$.

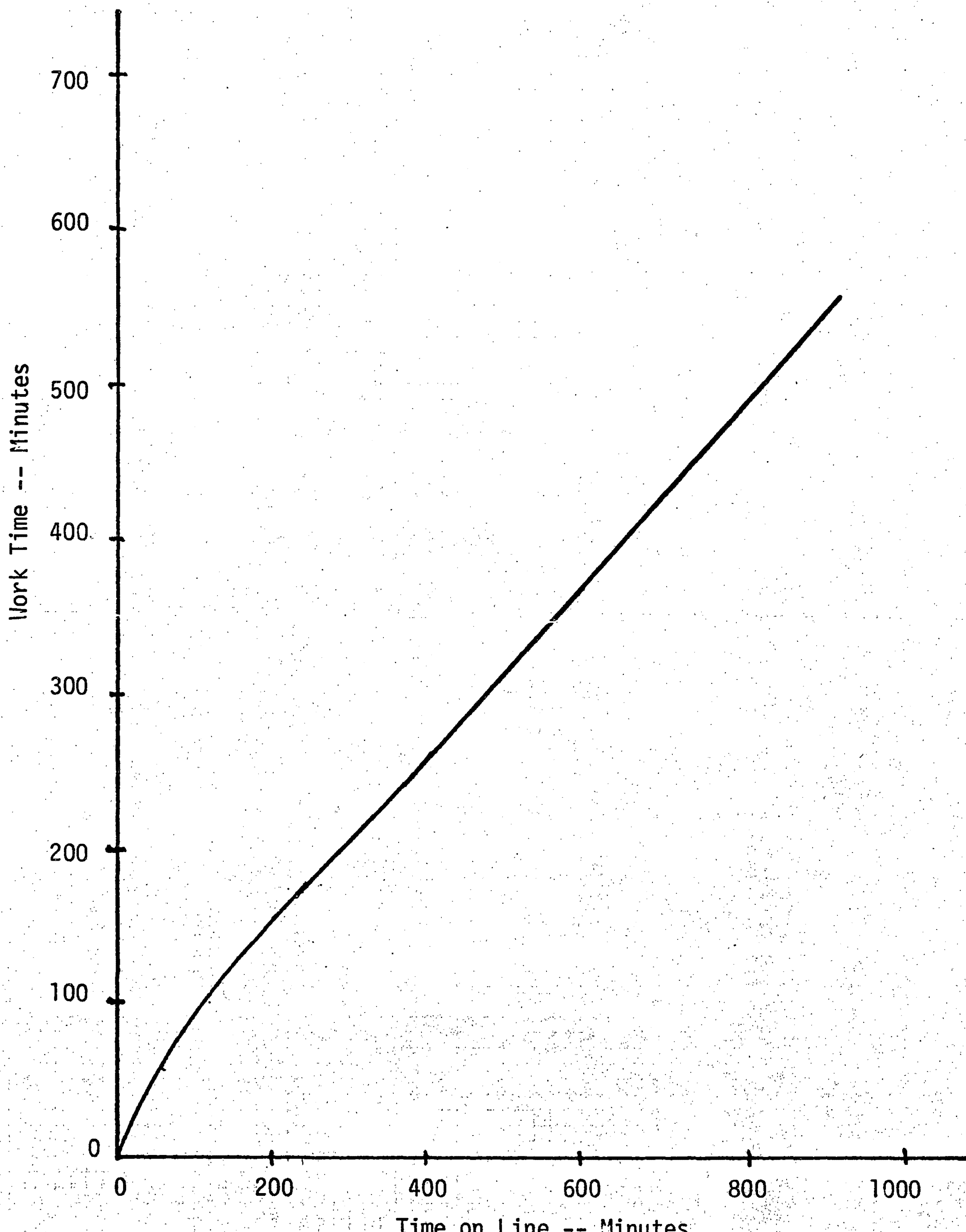
4. MODEL RESULTS

Figure 69 shows work time plotted versus time on line. It can be seen that during the first 100 minutes work time is about the same as time on line (as reported by Lindquist), while for longer periods work time is a monotonic decreasing fraction of time on line.

5. Flow Chart

Not required.

Figure 69 Work Time as a Function of Time on Line



```
      FUNCTION WKTIM(OLTIM)
C
C     THIS FUNCTION CALCULATES THE TIME PERSONNEL ACTUALLY WORK
C     AS A FUNCTION OF TIME SPENT CONTINUOUSLY ON LINE
C     VERSION 1.00      2/11/76
C
C     PROGRAMMER - J KEEFER
C
C     MODEL - J SANDERLIN
C
C     INPUT
C         OLTIM  - TIME SPENT ON LINE CONTINUOUSLY(MINUTES)
C     OUTPUT
C         WKTIM  - THE TIME SPENT ACTUALLY WORKING ON LINE(MINUTES)
C
C     COMPUTE THE ACTUAL WORK TIME
C
      OLT = OLTIM/60.0
      WKT = 0.96*(OLT**0.796)
      WKT = AMIN1(OLT,WKT)
      WKTIM = WKT*60.0
C
  900 RETURN
      END
```

47.0 INCREMENTAL WORK TIME (DWKTIM) Version 1.00 15 June 1975

1. PURPOSE

This function computes the incremental time personnel work as a function of the continuous on-line time, and the incremental time spent on-line.

2. ARGUMENTS

INPUT

DOLTIM - incremental time spent on line (min)
OLTIM - continuous on-line time (min)

OUTPUT

DWKTIM - the incremental work time (min)

3. COMPUTATION PROCEDURE

The work time (t_{w1}) at the start of the time increment is computed from,

$$t_{w1} = t_w(t_{o\ell}), \tag{1}$$

where $t_{o\ell}$ is the continuous on-line time. The work time (t_{w2}) at the end of the time increment is given by

$$t_{w2} = t_w(t_{o\ell} + \Delta t_{o\ell}), \tag{2}$$

where $\Delta t_{o\ell}$ is the incremental time on-line, and $t_w(t)$ is the actual work time function (see documentation for WKTIM).

The incremental work time is defined by

$$\Delta t_{o\ell} = t_{w2} - t_{w1}. \tag{3}$$

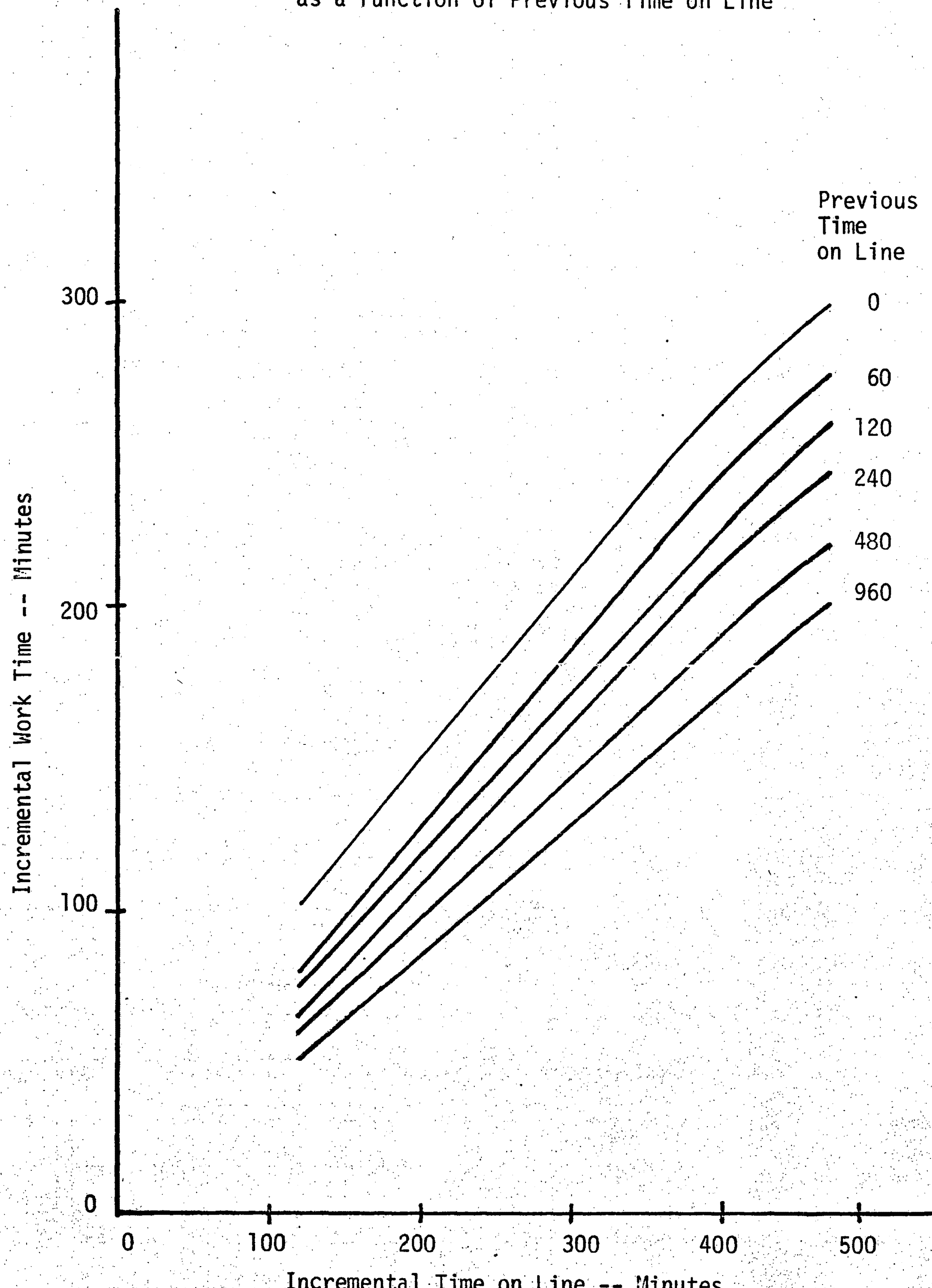
4. MODEL RESULTS

Figure 70 shows incremental work time plotted versus incremental time on line for several values of time spent on line previous to the current increment. Incremental work time is a monotonic decreasing fraction of incremental time on line for increasing intervals of time on line because of the increasing amount of rest time that is required.

5. FLOW CHART

Not required.

Figure 70 Incremental Work Time Versus Incremental Time on Line as a Function of Previous Time on Line



```
      FUNCTION DWKTIM(DOLTIM,OLTIM)
C
C     THIS FUNCTION CALCULATES THE INCREMENTAL TIME
C     PERSONNEL WORK.
C     VERSION 1.00      2/11/76
C
C     PROGRAMMER - J KEEFER
C
C     MODEL - J SANDERLIN
C
C         INPUTS
C     DOLTIM - INCREMENTAL TIME SPENT ON LINE
C     OLTIM  - CONTINUOUS ON LINE TIME(MINUTES)
C         OUTPUT
C     DWKTIM - INCREMENTAL ON LINE WORK TIME(MINUTES)
C
C         ROUTINES USED
C     WKTIM
C
C     COMPUTE THE INCREMENTAL TIME ACTUALLY SPENT ON LINE
C
      DWKTIM = WKTIM(OLTIM+DOLTIM) - WKTIM(OLTIM)
C
      RETURN
      END
```

1. PURPOSE

This routine calculates the fatigued work rate of personnel as a function of the unfatigued work rate and continuous time spent on-line.

2. ARGUMENTS

INPUTS

WRTE	-	unfatigued work rate (meter^2/min/man)
OLTIM	-	continuous on-line time (min)

OUTPUTS

WRFAT	-	the fatigued work rate (meter^2/min/man)

3. COMPUTATION PROCEDURE

The data of Reference 12 were used to model the effects of fatigue on personnel work rates. The fatigued work rate (W_f) is expressed in terms of the unfatigued work rate (W_o) and the applied work time (t_w) as follows;

$$W_f = W_o(1.0 - 0.137t_w + 0.005t_w^2), \quad (1)$$

where t_w is the applied work time. The applied work time is obtained from the actual work time function (see documentation for WKTIM routine), in terms of the continuous on-line time ($t_{o\ell}$) by

$$t_w = t_w(t_{o\ell}).$$

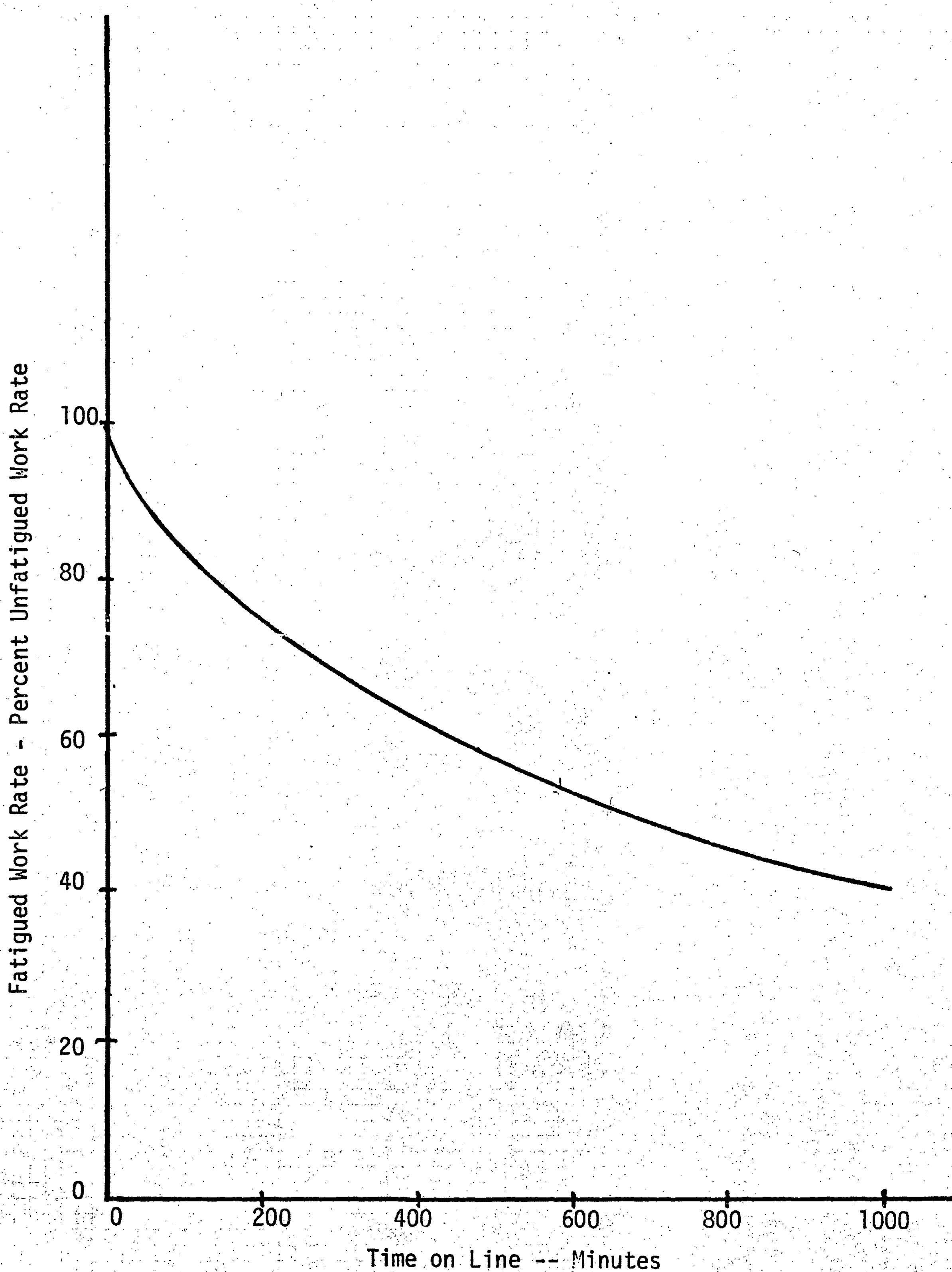
4. MODEL RESULTS

Figure 71 shows fatigued/unfatigued work rate plotted as a percent against time on line. It can be seen that, due to fatigue effects, percent work rate is a monotonic decreasing function of time on line. An unfatigued work rate would be represented in Figure 71 by a horizontal line at 100 percent.

5. FLOW CHART

Not required.

Figure 71 Effect of Fatigue on Work Rate as a Function of Time on Line



```
      FUNCTION WRFAT(WRTE,OLTIM)
C
C     WRFAT CALCULATES THE FATIGUED WORK RATE AS A FUNCTION
C     OF UNFATIGUED WORK RATES AND CONTINUOUS ON LINE TIME
C     VERSION 1.00      2/9/76
C
C     PROGRAMMER - J KEEFER
C
C     MODEL - J SANDERLIN
C
C         INPUTS
C     WRTE    - UNFATIGUED WORK RATE(METER SQ./MIN/MAN)
C     OLTIM   - CONTINUOUS TIME SPENT ON LINE(MINUTES)
C         OUTPUT
C     WRFAT   - FATIGUED WORK RATE(METER SQ./MIN/MAN)
C
C         ROUTINES USED
C     WKTIM
C
C     COMPUTE FATIGUED WORK RATE FROM ACTUAL TIME ON LINE
C
      TW = WKTIM(OLTIM)/60.0
      WRFAT = WRTE*(1.0 - 0.137*TW + 0.005*TW*TW)
C
      RETURN
      END
```

49.0 AVERAGE FATIGUED WORK RATES (WRFATA) Version 1.00 15 June 1975

1. PURPOSE

This routine calculates an average fatigued work rate over a specified time increment.

2. ARGUMENTS

INPUTS

WRTE	-	unfatigued work rate (meter^2/min/man)
OLTIM	-	continuous on-line time during a shift (min)
DOLTIM	-	incremental on-line time beyond OLTIM (min)

OUTPUTS

WRFATA	-	averaged fatigued work rate (meter^2/min/man)

3. PROCEDURE

The fatigued work rate, w_{f1}, at the start of the time increment is given by

$$w_{f1} = w_f(w_o, t_{o\ell}), \tag{1}$$

where w_o is the unfatigued work rate and $t_{o\ell}$ is the time spent on-line at the start of the time interval. The fatigued work rate, w_{f2}, at the end of the time increment is given by

$$w_{f2} = w_f(w_o, t_{o1} + \Delta t_{o\ell}), \tag{2}$$

where $\Delta t_{o\ell}$ is the incremental time spent on-line beyond $t_{o\ell}$. The function w_f is the fatigued work rate function (see documentation for WRFAT). The average fatigued work rate $\overline{W}_f$ is computed from

$$\overline{W}_f = 0.5(w_{f1} + w_{f2}). \tag{3}$$

4. FLOW CHART

Not required.

```
      FUNCTION WRFATA(WRTE,OLTIM,DOLTIM)
C
C     THIS FUNCTION CALCULATES AN AVERAGE FATIGUED
C     WORK RATE OVER A TIME INTERVAL
C     VERSION 1.00      2/11/76
C
C     PROGRAMMER - J KEEFER
C
C     MODEL - J SANDERLIN
C
C         INPUTS
C     WRTE   - UNFATIGUED WORK RATE(SQ.METER/MIN/MAN)
C     OLTIM  - CONTINUOUS ON LINE TIME (MINUTES)
C     DOLTIM - INCREMENTAL TIME ON LINE BEYOND OLTIM(MINUTES)
C         OUTPUT
C     WRFATA - AVERAGED FATIGUED WORK RATE(METER SQ./MIN./MAN)
C
C          ROUTINES USED
C     WRFAT
C
C     COMPUTE AVERAGED FATIGUED WORK RATE
C
      WR1 = WRFAT(WRTE,OLTIM)
      WR2 = WRFAT(WRTE,OLTIM+DOLTIM)
      WRFATA = 0.5*(WR1+WR2)
C
      RETURN
      END
```

50.0 LINE WIDTH ESTIMATE (CALWID) Version 1.00 9 January 1976

1. PURPOSE

This routine estimates required line width as a function of fuel load and wind speed.

2. ARGUMENTS

<u>INPUT</u>

WD - fuel load (KG/M^2)

WS - wind speed (M/Min)

<u>OUTPUT</u>

WIDTH - line width (m)

3. PROCEDURE

The required line width at zero wind speed is defined to be a function of fuel load of the form:

$$W_o = 0.723\ F_L - 6.66 \times 10^{-3} F_L^{\ 2} - 4.42 \times 10^{-4} F_L^{\ 3}$$

where

W_o = line width (ft.)

F_L = fuel load (tons/acre).

Line width as a function of wind speed is obtained from W_o by:

$$W(V_w) = W_o \exp(0.054 V_w)$$

where

$W(V_w)$ = line width (ft.)

V_w = wind speed (mph).

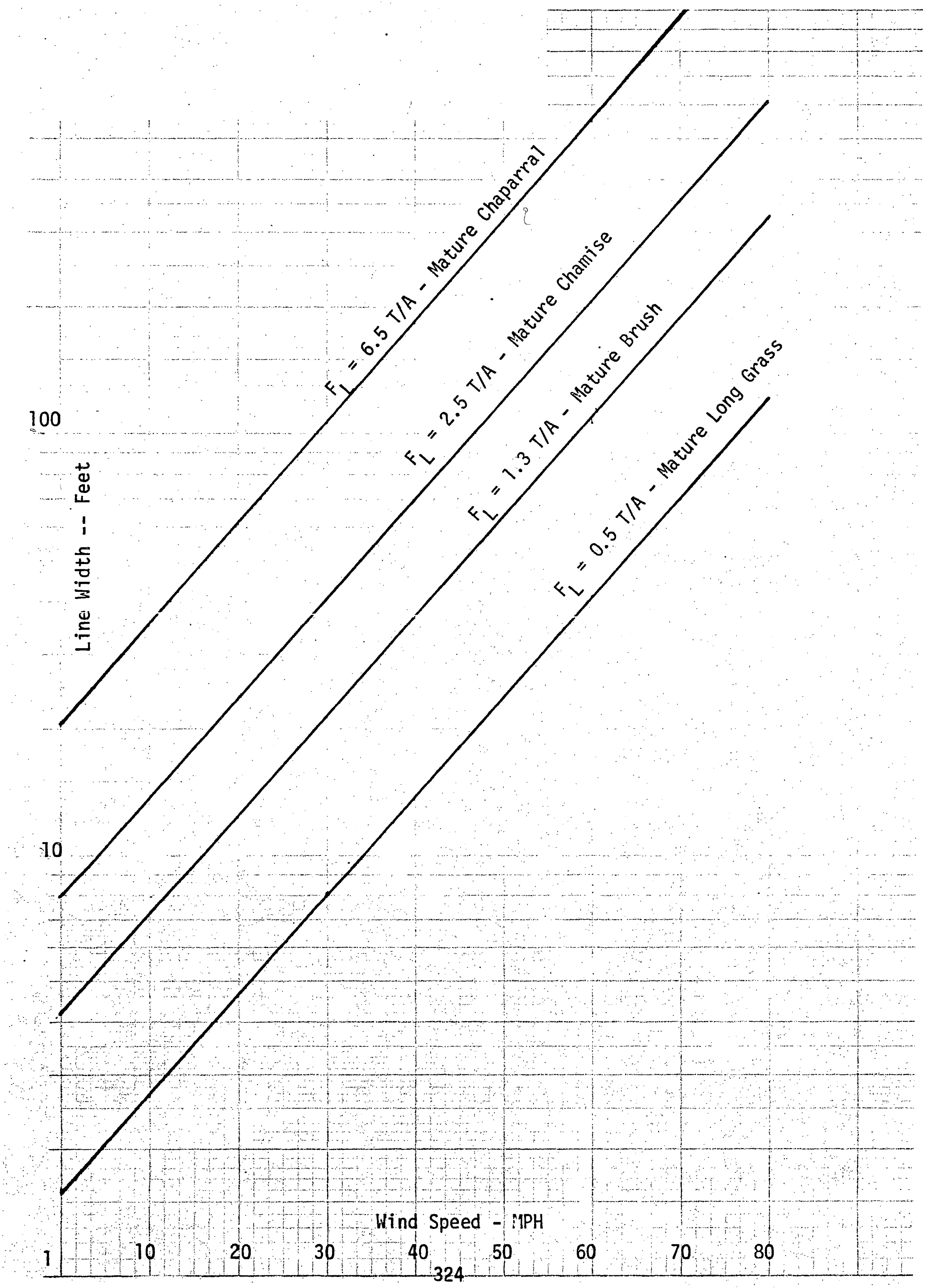
4. MODEL RESULTS

Figure 72 shows line width plotted against wind speed for 4 values of fuel load representative of 4 classes of mature fuels. It can be seen that the no wind line width (W_o) is approximately a linear function of fuel load, while the wind dependent line width is an exponential function of wind speed. The line widths for wind speeds up to about 30 mph are probably reasonable or possibly conservative; however, even a 500 ft. wide line would probably not contain a fire in 25 year old chaparral driven by a 60 mph wind.

5. FLOW CHART

Not required.

Figure 72 Line Width Versus Wind Speed with Fuel Load as a Parameter



```
      SUBROUTINE CALWID(WO,WS,WIDTH)
C
C     VERSION 1.00     1/9/76
C
C     THIS ROUTINE ESTIMATES LINE WIDTH AS A
C     FUNCTION OF FUEL LOADING AND WIND SPEED.
C
C     INPUTS
C          WO = TOTAL FUEL LOAD (KG/M**2)
C          WS = WIND SPEED (M/MIN)
C     OUTPUT
C          WIDTH = LINE WIDTH (METERS)
C
C     CONVERT WO TO FEET AND WS TO MPH
      FL=WO*4.46
      VW=WS*0.03726
C          WIDTH DUE TO FUEL LOAD
      WFL=0.723*FL-6.656E-3*FL**2-4.42E-4*FL**3
C          WIDTH MODIFICATION FOR WIND SPEED
      WFLWND=WFL*(1.0+(0.116*VW-8.33E-4*VW**2))
C          OUTPUT WIDTH IN METERS
      WIDTH=WFLWND*0.3048
      RETURN
      END
```

51.0 CALCULATION OF A DISCRETE DERIVATIVE (DERIV)

1. Purpose

This routine provides derivatives of functions described by a series of points.

2. Arguments

INPUTS

XSTEP	-	Increment in X between Y evaluations
X1	-	Value of X at YVAL(1)
YVAL	-	An array of function values to be differentiated (SIZE = NSTEP)
NSTEP	-	Number of values input

OUTPUTS

DYVAL	-	Discrete derivative of YVAL (an array, SIZE = NSTEP)
X10	-	Value of X at DYVAL(1)

3. Procedure

The derivative is calculated by the following difference equation:

$$DVAL_i = (YVAL_{i+1} - YVAL_i)/XSTEP$$

4. Comments

None.

5. Flow Chart

Not required.

```
      SUBROUTINE DERIV(XSTEP,X1,YVAL,NSTEP,DYVAL,X10)
C
C     THIS ROUTINE CALCULATES THE DERIVATIVE OF A FUNCTION
C     VERSION 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     XSTEP  - INCREMENT IN X BETWEEN Y EVALUATIONS
C     X1     - VALUE OF X AT YVAL (1)
C     YVAL   - AN ARRAY OF FUNCTION VALUES TO BE DIFFERENTIATED
C                (SIZE=NSTEP)
C     NSTEP  - NUMBER OF VALUES INPUT
C         OUTPUTS
C     DYVAL  - DESCRETE DERIVATIVE OF YVAL (AN ARRAY, SIZE=NSTEP-1)
C     X10    - VALUE OF X AT DYVAL(1)
C
      DIMENSION YVAL(NSTEP),DYVAL(NSTEP)
C
C     CALCULATE FIRST X VALUE FOR DYVAL (ONLY FOR SMALL XSTEP)
C
      X10=X1
C
C     CALCULATE DERIVATIVE
C
      DO 100 I=2,NSTEP
      IM1=I-1
      DYVAL(IM1)=(YVAL(I)-YVAL(IM1))/XSTEP
  100 CONTINUE
      DYVAL(NSTEP)=DYVAL(NSTEP-1)
C
      RETURN
      END
```

52.0 MAXIMUM PROBABILITY (MAXCR)

1. Purpose

This routine stores the maximum probability of control, together with the associated probability of containment and the time.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

The current value of probability of control is compared with a stored value and the largest of the two is stored.

4. Comments

None.

5. Flow Chart

Not required.

```
      SUBROUTINE MAXCR(CCR,CCN,CTIM,CRMAX,CNAMAX,TMAX)
C
C     THIS ROUTINE STORES THE MAXIMUM PROBABILITY OF CONTROL
C     AND THE ASSOCIATED PROBABILITY OF CONTAINMENT AND TIME
C     VERSION 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C            INPUTS
C     CCR      - CURRENT PROBABILITY OF CONTROL
C     CCN      - CURRENT PROBABILITY OF CONTAINMENT
C     CTIM     - CURRENT TIME
C            INPUT/OUTPUTS
C     CRMAX    - MAXIMUM VALUE OF PROBABILITY OF CONTROL (LAST FOUND)
C     CNAMAX   - PROBABILITY OF CONTAINMENT AT CRMAX
C     TMAX     - TIME AT CRMAX
C
      IF(CRMAX.GT.CCR)GO TO 10
      CRMAX=CCR
      CNAMAX=CCN
      TMAX=CTIM
   10 RETURN
      END
```

53.0 TIME REFORMATTING (OUTTIM)

1. Purpose

This routine converts a base date and relative time to output form.

2. Arguments

Each of the arguments is defined in the opening comments of the listing which follows.

3. Procedure

Two subroutines (CNRTST, CNDYMD) are used to perform the conversions. The hours and minutes are calculated from the new relative time generated by CNRTST.

4. Comments

None.

5. Flow Chart

Not required.

```
      SUBROUTINE OUTTIM(TOF,JDOF,IYRF,TIM,MONC,JDAYC,IYRC,IHRSC,IMINC)
C
C     THIS SUBROUTINE CONVERTS A BASE DATE AND RELATIVE TIME
C     TO OUTPUT FORM (WRITTEN ONLY TO AVOID REPETIOUS CODING)
C     VER 1.00
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C             INPUTS
C     TOF     - BASE TIME
C     JDOF    - BASE DAY OF YEAR
C     IYRF    - BASE YEAR
C     TIM     - RELATIVE TIME
C             OUTPUTS
C     MONC    - MONTH
C     JDAY    - DAY
C     IYRC    - YEAR
C     IHRSC   - HOURS (24 HOUR)
C     IMINC   - MINUTES
C
      CALL CNRTST(TOF,JDOF,IYRF,TIM,TIMC,IDAYC,IYRC)
      CALL CNDYMD(IDAYC,IYRC,MONC,JDAYC)
      IHRSC=IFIX(TIMC/60.0)
      IMINC=IFIX(TIMC-FLOAT(IHRSC*60)+0.5)
      RETURN
      END
```

54.0 TERMINAL PLOTTING (PLOTXY)

1. Purpose

This routine produces X-Y printer plots for functions, and produces enlargements of any portion of the plot.

2. Arguments

XARY	-	An array of X values*
YARY	-	An array of Y values*
NUMPTS	-	Number of points in XARY (YARY)
CHAR	-	Character to use as plot symbol (alpha)
LABELX	-	Label for X axis (string)
LABELY	-	Label for Y axis (string)
TITLE	-	Plot title (string)

3. Procedure

All pairs (S_i, Y_k) are checked to see if they are within each character location on the plot. If none is found the position is left blank, otherwise it is filled with the appropriate character.

4. Comments

None.

5. Flow Chart

See Figure 73.

---

*The last two values (NUMPTS + 1 and NUMPTS + 2) are the minimum value and increment, respectively. The plot is 60 characters by 60 characters (i.e., the maximum Y is 60 incr. + min Y).

Figure 73 Logic Structure for Routine PLOTXY

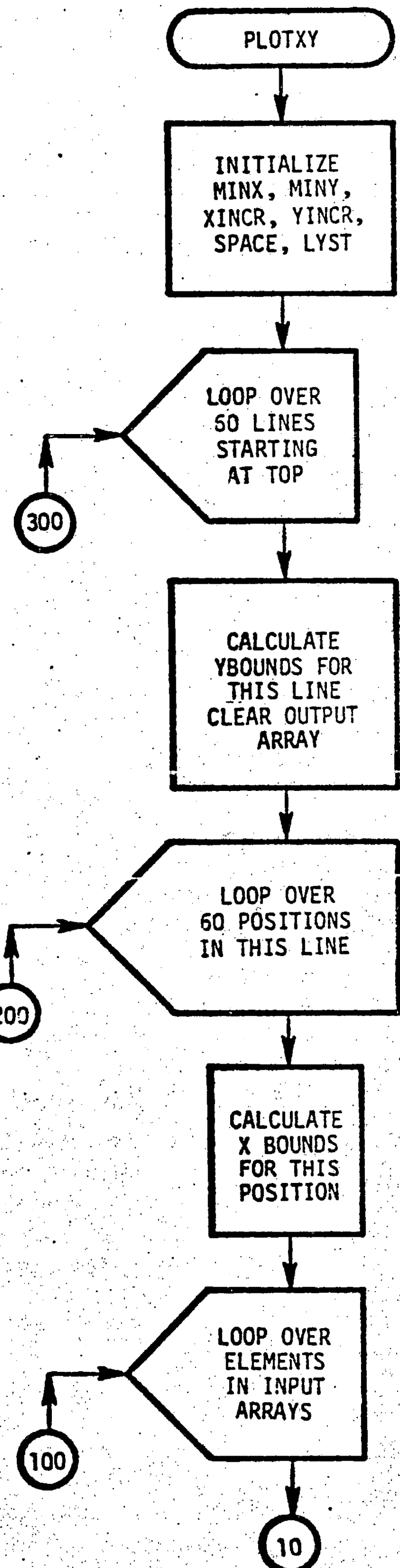


Figure 73 Logic Structure for Routine PLOTXY (cont'd)

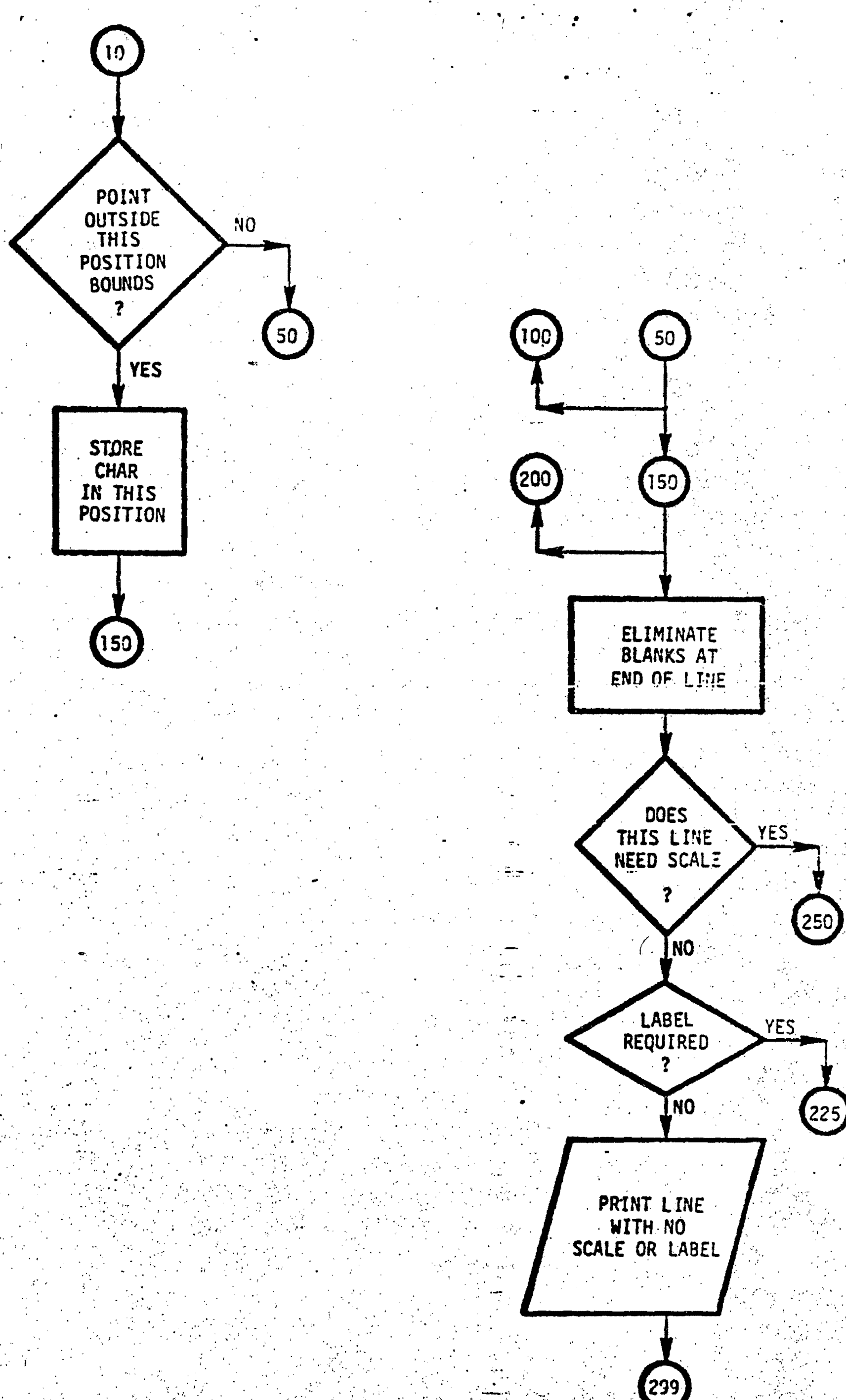


Figure 73 Logic Stucture of Routine PLOTXY (cont'd)

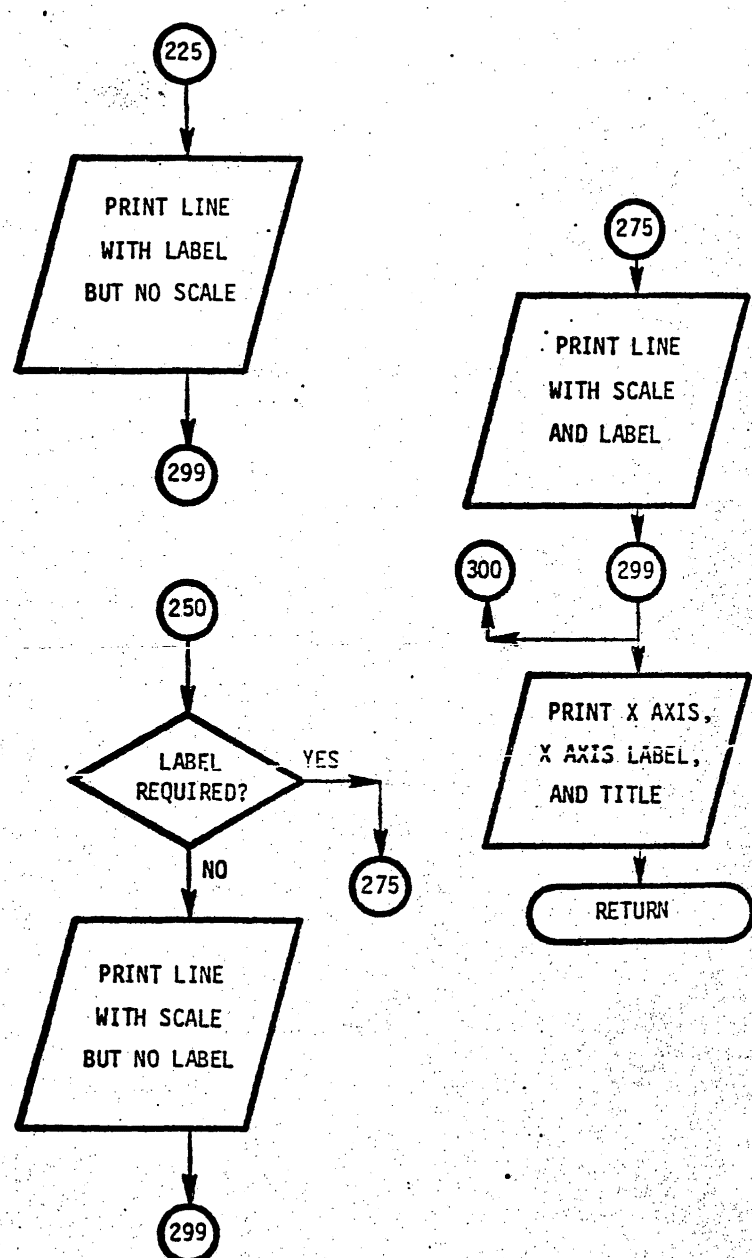
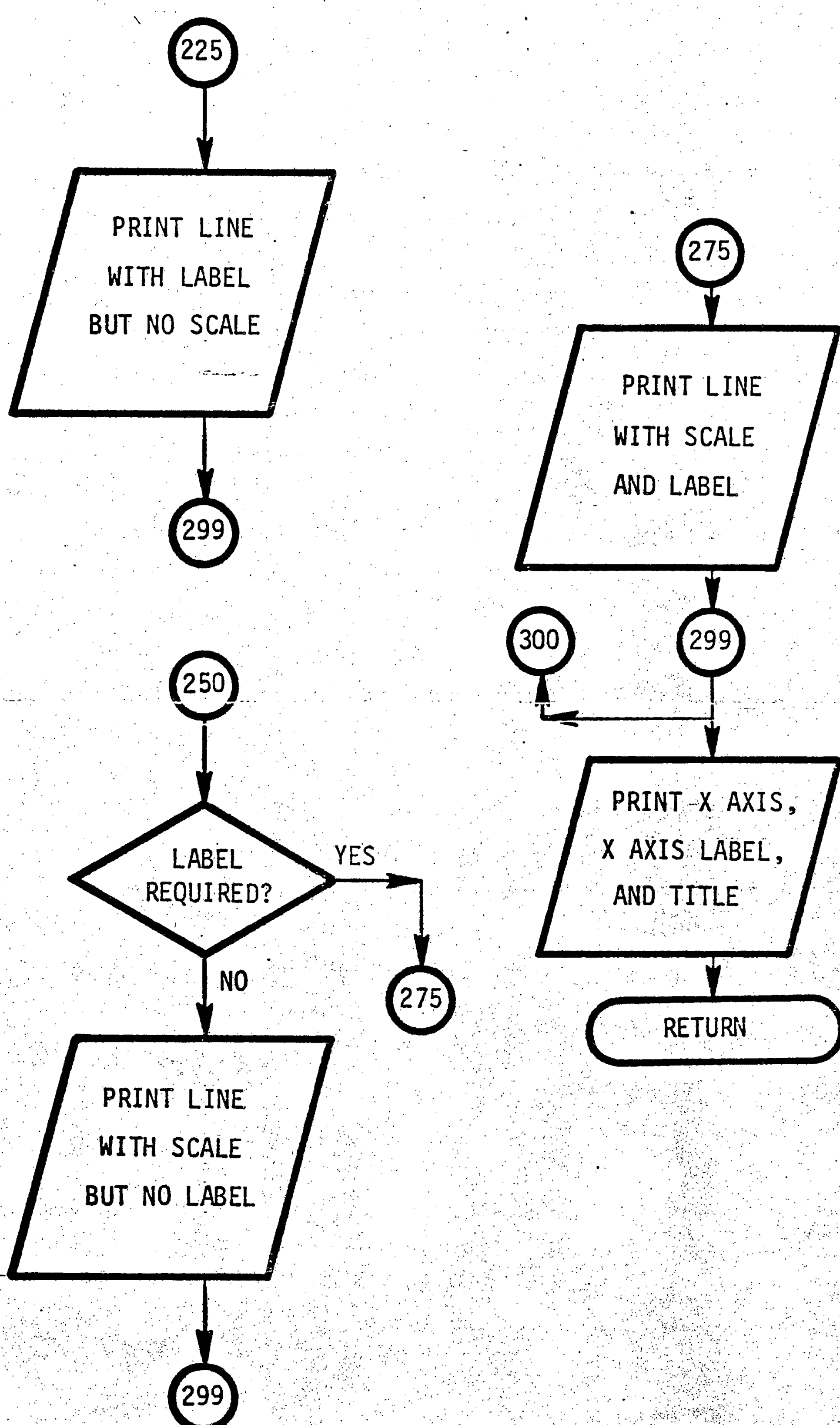


Figure 73 Logic Structure of Rountine PLOTXY (cont'd)



```
      SUBROUTINE PLOTXY(XARY,YARY,NUMPTS,CHAR,LABELX,LABELY,TITLE)
C
C     X-Y PLOT ROUTINE FOR EXECUPORT TERMINAL
C     VER 1.20
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS
C     XARY   - ARRAY OF X COORDINATES OF LINE TO BE PLOTTED
C     YARY   - ARRAY OF Y COORDINATES OF LINE TO BE PLOTTED
C     NUMPTS - NUMBER OF POINTS TO BE PLOTTED
C     CHAR   - ARRAY OF CHARACTERS TO BE USED AS PLOT SYMBOLS (ALPHA)
C     LABELX - LABEL FOR X AXIS (STRING)
C     LABELY - LABEL FOR Y AXIS (STRING)
C     TITLE  - PLOT TITLE (STRING)
C   * NOTE - THE LAST TWO VALUES (NUMPTS+1 AND NUMPTS+2) ARE THE
C     MINUMUM AND INCREMENT VALUES RESPECTIVELY.  THE MAXIMUM =
C     MINIMUM + (NUMPTS - 1) * INCREMENT.
C         OUTPUT
C     THIS ROUTINE PRODUCES A 5 X 5 INCH PLOT ON THE STANDARD PRINT
C     DEVICE.  IT USES STRING VARIABLES AND THUS WILL NEED
C     MODIFICATION TO CHANGE TO ANOTHER MACHINE.
C
C     NOTICE.......... FOR USE ON G. E. TIMESHARE ONLY
C
      DIMENSION XVOUT(6),XARY(100),YARY(100)
C
      STRING LABELX,LABELY,TITLE,LCHAR,SPACE,TEMP,NULL
      REAL MINY,MINX
      ALPHA ALPOUT(60),CHAR(100)
C
C  ***********************************
C
C
      NPTSV=30
      NPTSH=50
      NSTPV=NPTSV/6+1
C
      NULLIFY NULL
C         PRINT TITLE
      SPACE="                             "
      NLT=LENSTR(TITLE)
      NINS=NLT/2
      SPACE=ALTSTR(NULL,SPACE,1,NINS)
      TEMP=ALTSTR(SPACE,TITLE,0,0)
      PRINT    1008,TEMP
      SPACE="                             "
C
      MINX=XARY(NUMPTS+1)
      MINY=YARY(NUMPTS+1)
      XINCR=XARY(NUMPTS+2)*0.1
      YINCR=YARY(NUMPTS+2)*0.16666666667
C
      NLY=LENSTR(LABELY)
C
      LY=1
      LYST=NPTSV-(NPTSV-NLY)/2
      YMAX=FLOAT(NPTSV)*YINCR+MINY
      RINCR=10.0*XINCR
      DO 10 I=1,NSTPV
      XVOUT(I)=MINX+RINCR*FLOAT(I-1)
   10 CONTINUE
      PRINT    1007,XVOUT
 1007 FORMAT(//8X,7G10.3)
      PRINT    1004
      PRINT    1002 , YMAX
```

```
C
C         LOOP OVER Y
C
          DO 300 N=1,NPTSV
          L=(NPTSV+1)-N
C             CALCULATE YMIN,YMAX FOR THIS BLOCK
          YMAX=FLOAT(L)*YINCR+MINY
          YMIN=FLOAT(L-1)*YINCR+MINY
C             CLEAR OUTPUT ARRAY
          DO 20 I=1,NPTSH
          ALPOUT(I)="    "
     20   CONTINUE
C
C         LOOP OVER X
C
          DO 200 M=1,NPTSH
C             CALCULATE XMIN,XMAX FOR THIS BLOCK
          XMIN=FLOAT(M-1)*XINCR+MINX
          XMAX=FLOAT(M  )*XINCR+MINX
C
C         LOOP OVER INPUT ARRAY
C
          DO 100 J=1,NUMPTS
C             DETERMINE IF POINT IS WITHIN BLOCK
          IF(XARY(J).LT.XMIN.OR.XARY(J).GE.XMAX)GO TO 50
          IF(YARY(J).LT.YMIN.OR.YARY(J).GE.YMAX)GO TO 50
          ALPOUT(M)=CHAR(J)
          GO TO 150
     50   CONTINUE
    100   CONTINUE
    150   CONTINUE
    200   CONTINUE
C
C         PRINT LINE WITH OR WITHOUT LABELING AS REQUIRED
C
C             ELIMINATE BLANKS ON END OF LINE
          DO 210 ISC=1,NPTSH
          ILJ=(NPTSH+1)-ISC
          IF(ALPOUT(ILJ).NE." ")GO TO 212
    210   CONTINUE
    212   CONTINUE
C             DOES THIS LINE NEED SCALE
          IF(MOD(N,6).EQ.0)GO TO 250
C             THIS LINE DOES NOT REQUIRE SCALE
C             DOES THIS LINE REQUIRE LABEL
          IF(L.LE.LYST.AND.LY.LE.NLY)GO TO 225
C             NO LABEL, NO SCALE
          PRINT    1000,(ALPOUT(IOUT),IOUT=1,ILJ)
   1000   FORMAT(15X,1HX,60A1)
          GO TO 299
C             LABEL, NO SCALE
    225   LCHAR=EXTSTR(LABELY,LY,1)
          PRINT    1001,LCHAR,(ALPOUT(IOUT),IOUT=1,ILJ)
   1001   FORMAT(2X,1A1,12X,1HX,60A1)
          LY=LY+1
          GO TO 299
C             THIS LINE REQUIRES SCALES
C             DOES THIS LINE REQUIRE LABEL
    250   IF(L.LE.LYST.AND.LY.LE.NLY)GO TO 275
C             NOLABEL, SCALE
          PRINT    1002,YMIN,(ALPOUT(IOUT),IOUT=1,ILJ)
   1002   FORMAT(4X,G10.3,2H I,60A1)
          GO TO 299
C             LABEL, SCALE
    275   LCHAR=EXTSTR(LABELY,LY,1)
          PRINT    1003,LCHAR,YMIN,(ALPOUT(IOUT),IOUT=1,ILJ)
   1003   FORMAT(2X,1A1,1X,G10.3,2H I,60A1)
          LY=LY+1
```

```
C          NEXT LINE
  299 CONTINUE
  300 CONTINUE
C          PRINT BOTTOM AND X LABEL
      NLX=LENSTR(LABELX)
      NINS=NLX/2
      NULLIFY NULL
      SPACE = ALTSTR(NULL,SPACE,1,NINS)
      TEMP=ALTSTR(SPACE,LABELX,0,0)
      PRINT    1004
 1004 FORMAT(15X,1HX,5(10HIXXXXXXXXX),1HI)
      PRINT     1005,XVOUT,TEMP
 1005 FORMAT(8X,6G10.3/16X,A//)
C          PRINT TITLE
      SPACE="                                 "
      NLT=LENSTR(TITLE)
      NINS=NLT/2
      SPACE=ALTSTR(NULL,SPACE,1,NINS)
      TEMP=ALTSTR(SPACE,TITLE,0,0)
      PRINT     1006,TEMP
 1006 FORMAT(16X,A//)
      NULLIFY TEMP,SPACE
C
  350 RETURN
 1008 FORMAT(//16X,A)
      END
```

55.0 TOPOGRAPHIC DATA BASE GENERATION PROGRAM (TDBGP)

1. Purpose

This program generates a topographic data base by statistical means for the Experimental Initial Attack Evaluation Program.

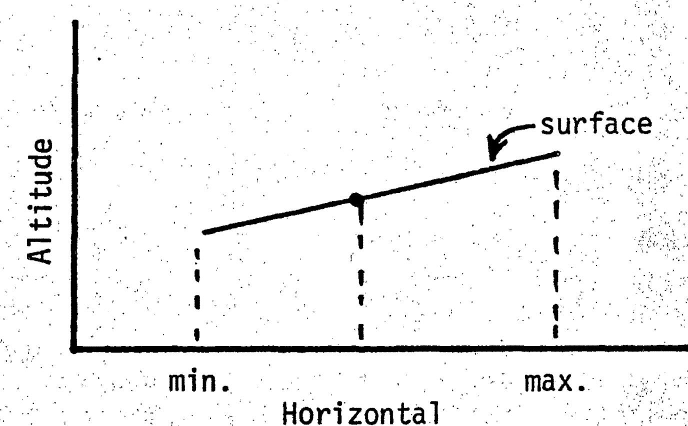
2. Input Requirements

Inputs are interactively obtained from the user. For a complete description see "User's Guide: Experimental Initial Attack Evaluation Model," MRC Report #7611-1-376, March 1976.

3. Procedure

The required inputs are obtained from the interactive user, the data base is generated and the results are written to the user selected file.

Data are generated on grids of two resolutions: coarse and fine. Data are generated on an X-Y coordinate grid, but for clarity the following figures will show the concepts in only one dimension. The boundaries in the horizontal direction, the altitude at the center and the slope(s) are used to begin the calculation. These data may be visualized as shown below.



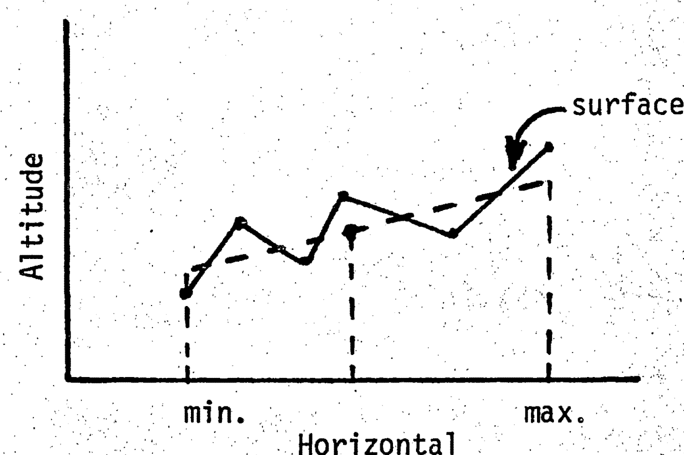
The slanting line represents a flat surface at the desired general slope. The following formulas are used to superimpose a coarse grid of large scale variation on the above flat surface.

HIXY = HBARC+(X-XC)*SLOPEX+(Y-YC)*SLOPEY
HC(I,J) = RANGAU(HIXY,DELH)

where

HIXY	=	mean altitude of flat surface at the point X,Y
HBARC	=	mean altitude at center of grid.
XC,YC	=	coordinates of grid center
X,Y	=	coordinates of a point X,Y
SLOPEX,SLOPEY	=	slope in the X and Y directions, respectively
RANGAU	=	a function which generates a gaussian distribution of random numbers.
DELD	=	large scale variation
HC	=	the course grid

After the above transformation the surface may be visualized as follows:



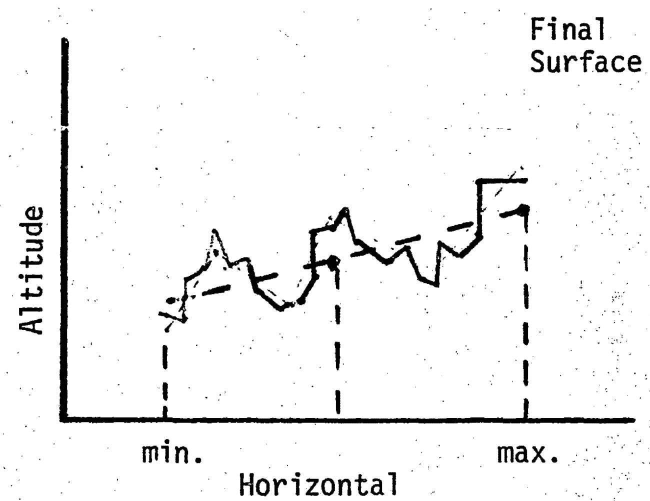
To achieve fine scale roughness the process is repeated. At each point a mean altitude of the coarse surface is first calculated by double linear interpolation. The following formula is then used to calculate the final grid of data.

HBC(I,J) = RANGAU(HBCMXY,DLH)

where

HBC	=	the fine grid
HBCMXY	=	mean altitude of the coarse surface
DLH	=	fine scale variation

The resulting surface may be visualized as follows.



4. Comments

None.

5. Flow Chart

See Figure 74.

Figure 74   Topographic Data Base Generation

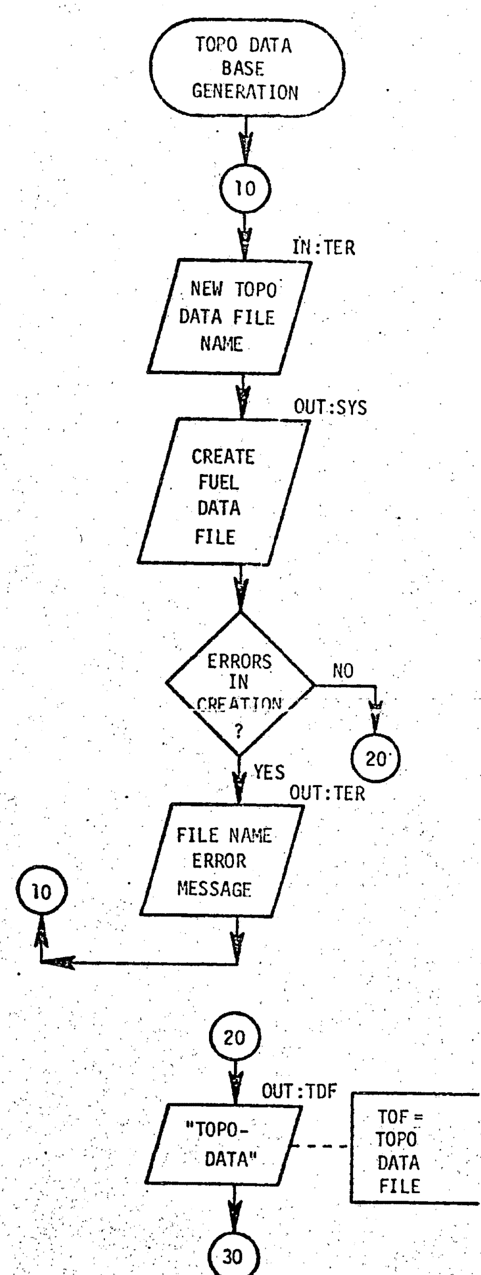


Figure 74 Topographic Data Base Generation (Cont'd)

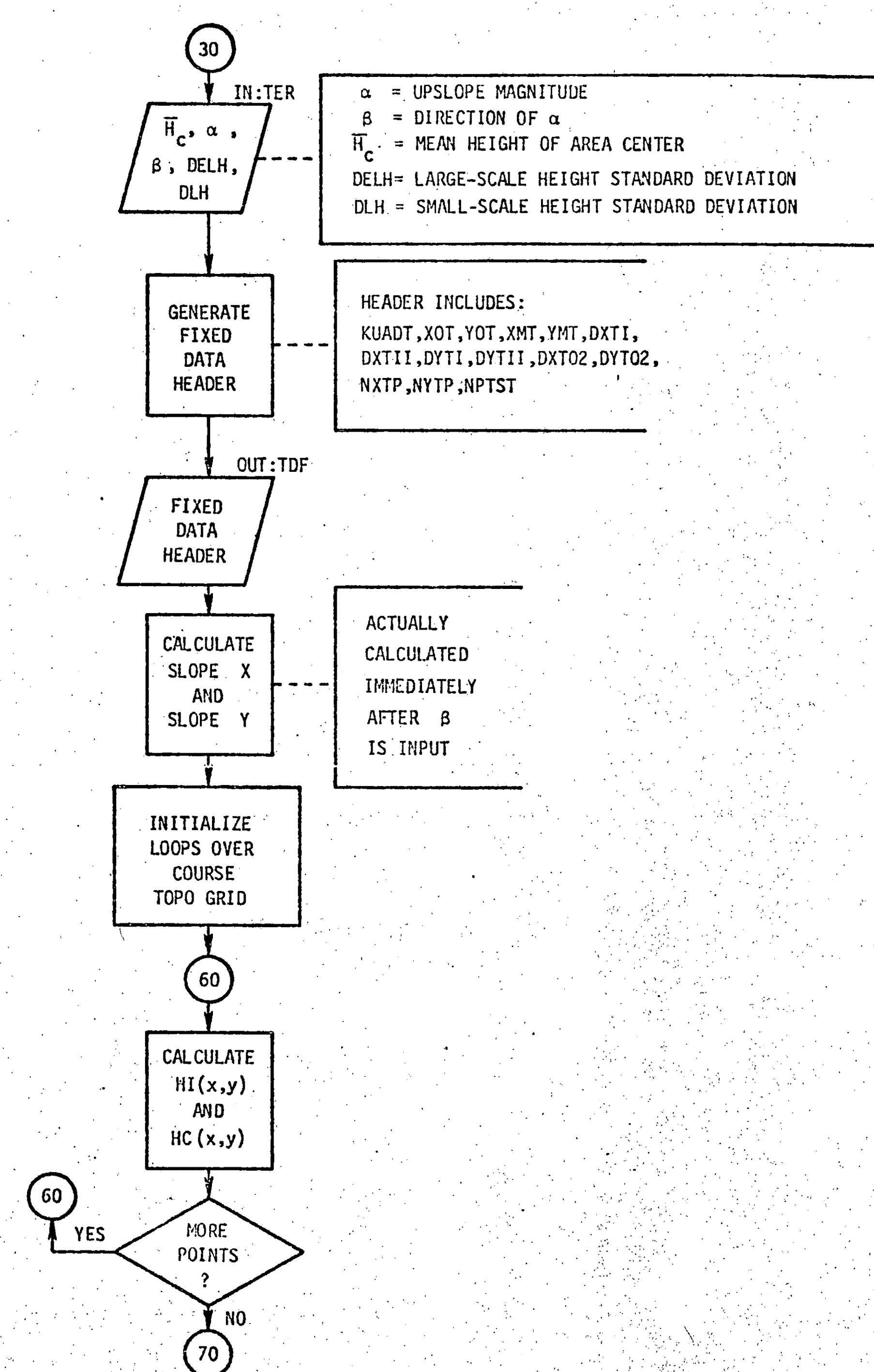
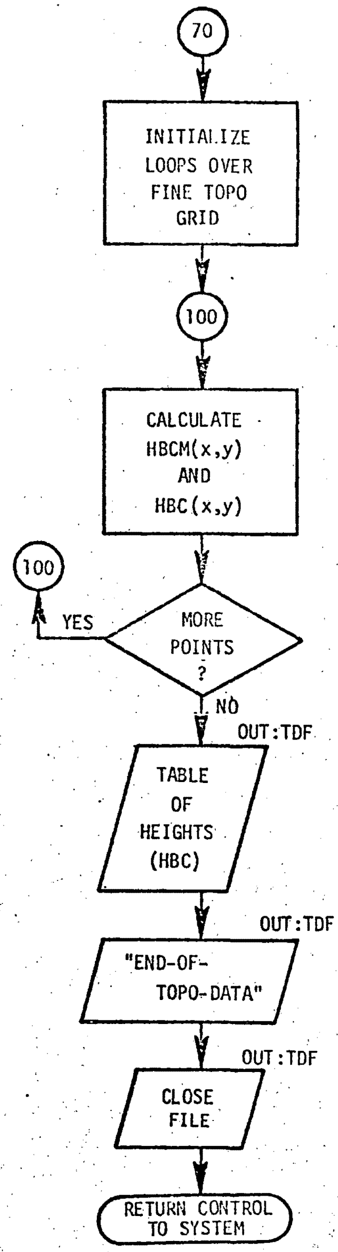


Figure 74 Topographic Data Base Generation (Cont'd)



$$$$$STDBGP

```
      FILELIST TDF
C
C     TOPO DATA BASE GENERATION PROGRAM
C     VER 1.00 ( DESIGNED FOR USE ON G. E. TIME SHARE)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS FROM TERMINAL
C      1.  FILE NAME FOR TOPO DATA FILE BEING GENERATED
C      2.  MEAN HEIGHT AT CENTER OF AREA
C      3.  UPSLOPE MAGNITUDE
C      4.  DIRECTION OF UPSLOPE FROM NORTH
C      5.  LARGE-SCALE HEIGHT STANDARD DEVIATION
C      6.  SMALL-SCALE HEIGHT STANDARD DEVIATION
C         OUTPUTS TO TOPO DATA FILE (TDF) IN ORDER OUTPUT
C     LABEL  - "TOPO DATA"
C     HEADER - CONTAINS ARRAY SIZES, COORDINATES, ETC.
C     TOPO TABLE
C     TRAILER LABEL - "END-OF-TOPO-DATA"
C         ROUTINES USES
C     CREATE-SYS, RANGAU
C
      DIMENSION HC(6,6),HHC(51,51)
C
C     IDENTIFY PROGRAM AND OPERATIONS TO BE PERFORMED
C
      CALL SETCHR(32)
      PRINT    200
  200 FORMAT(1X,44HTOPO DATA BASE GENERATION PROGRAM - VER 1.00/)
C
C     INPUT FROM TERMINAL - NEW TOPO DATA FILE NAME
C
   10 PRINT    201
  201 FORMAT(3X,29HENTER NEW TOPO DATA FILE NAME)
C
      PRINT    202
  202 FORMAT(7X,4H....)
      READ     210,TDF
  204 FORMAT(A8)
C
C     CREATE TOPO DATA FILE
C
      CALL CREATE (TDF," ",0,ISTAT)
      IF(ISTAT.EQ.0)GO TO 20
      PRINT    206
  206 FORMAT(3X,36H*---ERROR---*  INVALID NEW FILE NAME)
      GO TO 10
```

```
C
C      OUTPUT LABEL ON FILE TDF
C
    20 WRITE(1,207)
   207 FORMAT(9HTOPO-DATA)
C
C      INPUT FROM TERMINAL THE TOPO CHARACTERISTICS REQUIRED
C
    30 PRINT    209
   209 FORMAT(3X,43HENTER MEAN HEIGHT AT CENTER OF AREA IN FEET)
       PRINT    202
       READ     210,HBARC
       HBARC=HBARC*0.3048
   210 FORMAT(V)
C
       PRINT    212
   212 FORMAT(3X,30HENTER GENERAL SLOPE IN PERCENT)
       PRINT    202
       READ     210,ALPHA
       ALPHA=57.29577951*ATAN(ALPHA*0.01)
       IF(ALPHA.NE.0.0)GO TO 32
       SLOPEX=0.0
       SLOPEY=0.0
       GO TO 39
    32 PRINT    214
   214 FORMAT(3X,43HENTER DIRECTION OF GENERAL SLOPE FROM NORTH,
     & 10H (DEGREES))
       PRINT    202
       READ     210,BETA
       BETA=BETA*0.0174532925
       BETA=AMOD(BETA,6.2831853)
       SLOPEX=ALPHA*SIN(BETA)
       SLOPEY=ALPHA*COS(BETA)
C          CONTINUE INPUT
    39 PRINT    216
```

```
  216 FORMAT(3X,34HENTER LARGE-SCALE DEVIATION (FEET))
      PRINT    202
      READ     210,DELH
      DELH=DELH*0.3048
C
      PRINT 218
  218 FORMAT(3X,34HENTER SMALL-SCALE DEVIATION (FEET))
      PRINT    202
      READ     210,DLH
      DLH=DLH*0.3048
C
C     GENERATE FIXED DATA HEADER
C
      KUADT=1
      XOT=0.0
      YOT=0.0
      XMT=5000.0
      YMT=5000.0
      DXTI=100.0
      DXTII=0.01
      DYTI=100.0
      DYTII=0.01
      DXT02=50.0
      DYT02=50.0
      NXTP=51
      NYTP=51
      NPTST=260
      XC=2500.0
      YC=2500.0
      SRUF=DLH
C
C     OUTPUT FIXED DATA HEADER
C
      WRITE(1,220)KUADT,XOT,YOT,XMT,YMT,DXTI,DXTII,DYTI,DYTII,SRUF,
     &              DXT02,DYT02,NXTP,NYTP,NPTST
  220 FORMAT(I5,4E15.7/4E15.7/3E15.7,2I5,I10)
C
C     INITIALIZE LOOPS OVER COURSE TOPO GRID
C
      Y=YOT
      DO 60 J=1,6
      X=XOT
      DO 50 I=1,6
C         CALCULATE COURSE GRID
      H1XY=HBARC+(X-XC)*SLOPEX+(Y-YC)*SLOPEY
      HC(I,J)=RANGAU(H1XY,DELH)
C
      X=X+DXTI*10.0
   50 CONTINUE
      Y=Y+DYTI*10.0
   60 CONTINUE
C
C     INITIALIZE LOOPS OVER FINE GRID AND PREPARE FOR INTERPOLATION
C
      DO 100 J1=1,5
       J2=J1+1
      DO 90 JJ=1,11
       J3=(J1-1)*10+JJ
       DELY=FLOAT(JJ-1)*DYTI
       FRAC=DELY*DYTII
C
       DO 80 I1=1,5
        I2=I1+1
C
        Z1=HC(I1,J1)
        Z2=HC(I1,J2)
```

```
            Z3=HC(I2,J2)
            Z4=HC(I2,J1)
C
           DO 70 II=1,11
            I3=(I1-1)*10+II
            DELX=FLOAT(II-1)*DXTII
C
C      DOUBLE LINEAR LINEAR INTERPOLATION
C
       H1=(Z2-Z1)*FRAC+Z1
       H2=(Z3-Z4)*FRAC+Z4
C
       HBCMXY=H1+(H2-H1)*DELX*DXTII
C
C      ADD FINE ROUGHNESS
C
       HRC(I3,J3)=RANGAU(HBCMXY,DLH)
C
   70  CONTINUE
   80  CONTINUE
   90  CONTINUE
  100  CONTINUE
C
C      OUTPUT TOPO TABLE
C
       WRITE(1,222)HBC
  222  FORMAT(5E16.8)
C
C      OUTPUT TRAILER LABEL
C
       WRITE(1,224)
  224  FORMAT(16HEND-OF-TOPO-DATA)
C
       CLOSEFILE TDF
C
       PRINT    226
  226  FORMAT(1X,24HTOPO GENERATION COMPLETE)
C
       CALL EXIT
       STOP
C
C
       END
```

56.0 FUEL DATA BASE GENERATION PROGRAM (FDBGP)

1. Purpose

This program generates a fuel data base by statistical means for the Experimental Initial Attack Evaluation Program.

2. Input Requirements

Inputs are interactively obtained from the user. For a complete description see "User's Guide: Experimental Initial Attack Evaluation Model," MRC Report # 7611-1-376, March 1976.

3. Procedure

The required inputs are interactively obtained from the user, the data base is generated and the results are written to the user selected file.

Data are generated for fuel age variation only. The ages are generated using the following formula:

AGET(X,Y) = RANGAU(AGE,AGEV)

where

AGET	=	the age table being generated
AGE	=	the mean age
AGEV	=	the age variation
RANGAU	=	a function which generates a series of random numbers with gaussian distribution
X,Y	=	coordinates of point X,Y

4. Comments

None.

5. Flow Chart

See Figure 75.

Figure 75 Fuel Data Base Generation

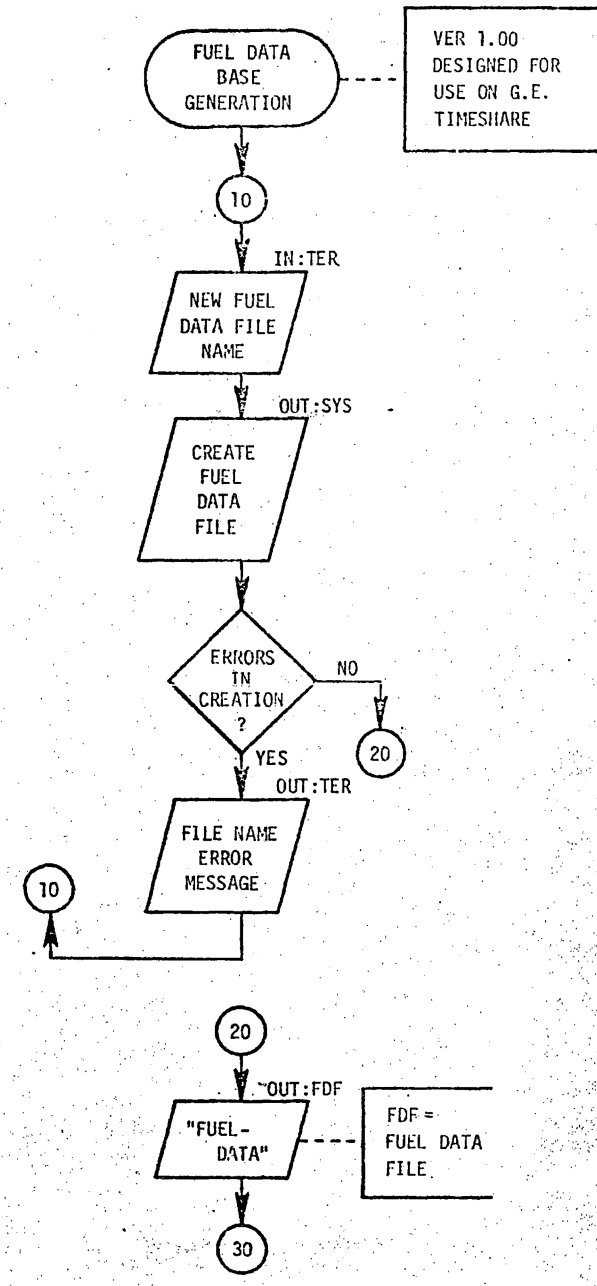


Figure 75 Fuel Data Base Generation (Cont'd)

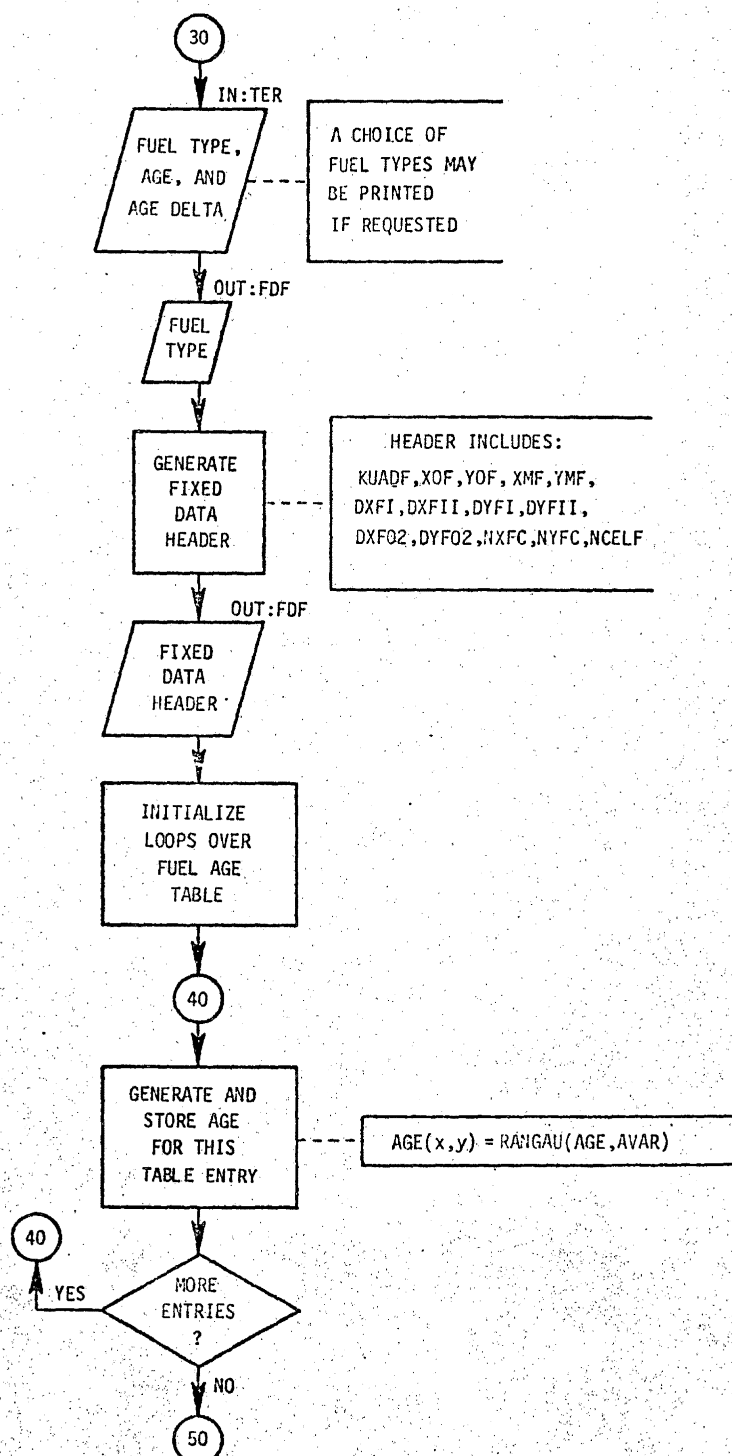
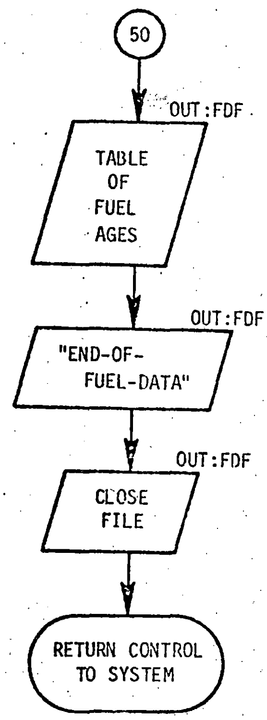


Figure 75 Fuel Data Base Generation (Cont'd)



```
$$$$$SFDBGP
      FILELIST FDF
C
C     FUEL DATA BASE GENERATION PROGRAM
C     VER 1.00 (DESIGNED FOR USE ON G. E. TIMESHARE)
C
C     PROGRAMMER - JOHN SUNDERSON, JR.
C
C         INPUTS FROM TERMINAL
C      1. FILE NAME FOR FUEL DATA FILE BEING GENERATED
C      2. FUEL TYPE INDEX (A CHOICE IS PRINTED IF REQUIRED)
C      3. FUEL AVERAGE AGE
C      4. FUEL AGE VARIATION
C         OUTPUTS TO FUEL DATA FILE (FDF) IN ORDER OUTPUT
C     LABEL  - "FUEL-DATA"
C     FUEL TYPE
C     HEADER - CONTAINS ARRAY SIZES, COORDINATES, ETC.
C     FUEL AGE TABLE
C     TRAILER LABEL - "END-OF-FUEL-DATA"
C         ROUTINES USED
C     CREATE-SYS, RANGAU
C
      DIMENSION AGET(50,50)
C
C     IDENTIFY PROGRAM AND OPERATIONS TO BE PERFORMED
C
      CALL SETCHR(32)
      PRINT    100
  100 FORMAT(1X,44HFUEL DATA BASE GENERATION PROGRAM - VER 1.00/)
C
C     INPUT FROM TERMINAL - NEW FUEL DATA FILE NAME
C
   10 PRINT    101
  101 FORMAT(3X,29HENTER NEW FUEL DATA FILE NAME  )
      PRINT    102
  102 FORMAT(7X,4H....)
      READ     109,FDF
C
C     CREATE FUEL DATA FILE
C
      CALL CREATE (FDF," ",0,ISTAT)
      IF(ISTAT.EQ.0)GO TO 20
      PRINT    106
  104 FORMAT(A8)
  106 FORMAT(3X,35H*---ERROR---* INVALID NEW FILE NAME)
      GO TO 10
C
C     OUTPUT LABEL ON FILE FDF
C
   20 WRITE(1,107)
  107 FORMAT(9HFUEL-DATA)
C
C     GET FUEL TYPE AND AGE CHARACTERISTICS FROM OPERATOR
C
   30 PRINT    108
  108 FORMAT(3X,39HENTER FUEL TYPE, AGE, AND AGE VARIATION/
     &         3X,10H(IN YEARS)/
     &         3X,44HZERO FOR TYPE PRINTS CHOICES (AGES IN YEARS))
      PRINT    102
      READ     109,IFTYPE,AGE,AGEV
  109 FORMAT(V)
      IF(IFTYPE.GT.5)GO TO 32
      IF(IFTYPE.NE.0)GO TO 35
   32 PRINT    110
  110 FORMAT(3X,23HPOSSIBLE FUEL TYPES ARE//
     &         3X,13H1-SHORT GRASS/
```

```
     &         3X,12H2-LONG GRASS/
     &         3X,23H3-BRUSH (NON-CHAPARRAL)/
     &         3X,22H4-CHAMISE (PURE STAND)/
     &         3X,17H5-MIXED CHAPARRAL)
      GO TO 30
C
C     OUTPUT FUEL TYPE TO FILE FDF
C
   35 WRITE(1,112)IFTYPE
  112 FORMAT(I5)
C
C     GENERATE FIXED DATA HEADER
C
      KUADF=1
      XOF=0.0
      YOF=0.0
      XMF=5000.0
      YMF=5000.0
      DXFI=100.0
      DXFII=0.01
      DYFI=100.0
      DYFII=0.01
      DXFO2=50.0
      DYFO2=50.0
      NXFC=50
      NYFC=50
      NCELF=2500
C
C     OUTPUT FIXED DATA HEADER
C
      WRITE(1,114)KUADF, XOF,YOF,XMF,YMF, DXFI,DXFII,DYFI,DYFII,
     &               DXFO2,DYFO2, NXFC,NYFC,NCELF
  114 FORMAT(I5,4E15.7/ 4E15.7/ 2E15.7,2I5,I10)
C
C
C     LOOP OVER ALL CELLS
C
C
      DO 45 J=1,NYFC
      Y=DYFO2 + FLOAT(J-1)*DYFI
      DO 40 I=1,NXFC
      X=DXFO2+FLOAT(I-1)*DXFI
C
C     GENERATE AND STORE AGE FOR A TABLE ENTRY
C
      AGET(I,J)=ABS(RANGAU(AGE,AGEV))
C
   40 CONTINUE
   45 CONTINUE
C
C     OUTPUT FUEL AGE TABLE
C
      WRITE(1,116)AGET
  116 FORMAT(5E16.8)
C
C     OUTPUT TRAILER LABEL AND CLOSE FDF
C
      WRITE(1,118)
  118 FORMAT(16HEND-OF-FUEL-DATA)
C
      CLOSEFILE FDF
C
      PRINT    120
  120 FORMAT(1X,24HFUEL GENERATION COMPLETE)
C
      CALL EXIT
      STOP
C
C
      END
```

<u>REFERENCES</u>

1. Sanderlin, James C. and Sunderson, John M., "Users Guide: Experimental Initial Attack Evaluation Model," Mission Research Corporation Report No. 7611-1-376, March 1976.

2. Hamming, R.W., <u>Numerical Methods for Scientists and Engineers</u>, McGraw-Hill Book Co., Inc., New York, N.Y., p. 34 and 389.

3. McCutchan, M.H., Ryan, B.C. and Schroeder, M.J., "Meteorlogical Input to Fire Command and Control System (FCCS) - Preliminary Report," February 1973.

4. Rothermel, Richard C., "A Mathematical Model for Predicting Fire Spread in Wildland Fuels," USDA Forest Service Research Paper INT-115, January 1972.

5. Rothermel, Richard C., and Philpot, Charles W., "Fire in Wildland Management - Predicting Changes in Chaparral Flammability," <u>Journal of Forestry</u>, Vol. 71, No. 10, October 1973.

6. Deeming, John E., "Fuel Modeling: the Key to Applying Rothermel's Mathematical Fire Spread Model to Fire-Danger Rating," Presented at the 1973 Spring Meeting of the Western States Section of <u>the Combustion</u> Institute, April 16, 1973.

7. Hanes, Ted L., "The Vegetation Called Chaparral," Department of Biological Sciences, California State University, Fullerton, California, 92634.

8. Countryman, Clive M., and Philpot, Charles W., "Physical Characteristics of Chamise as a Wildland Fuel," USDA Forest Service Research Paper PSW-66, 1970.

9. Countryman, Clive M., "This Humidity Business: What it is all about and its use in fire control," Pacific Southwest Forest and Range Experiment Station; USDA Forest Service, 1971.

10. Rothermel, Richard C., and Hardy, Charles E., "Influences of Moisture on Effectiveness of Fire Retardants," Intermountain Forest and Range Experiment Station, USDA Forest Service Research Paper INT-18, 1965.

11. Fosberg, Michael A. and Deeming, John A., "Derivation of the 1- and 10-Hour Timelag Fuel Moisture Calculations for Fire Danger Rating," USDA Forest Service Research Note RM-207, 1971.

12. Lindquist, James L., "Building Firelines - How Fast Do Crews Work?", Fire Technology, 11 June 1970.